COMMODORE 64

APPRENDRE À PROGRAMMER



DANS DES ACTIVITÉS DIRIGÉES

VOLUME I

LAURIAN PICARD

Le MINISTERE de l'EDUCATION du QUEBEC expose, .
de façon détaillée, le programme d'études au SECONDAIRE
de la science de l'informatique dans le **DOCUMENT 16-0081:**
"INTRODUCTION à la SCIENCE de l'INFORMATIQUE" (ISI)

Les VOLUMES 1 et 2 de cette série consacrée au **C64**
sont conformes au document ministériel et le VOLUME 2
contiendra le texte du programme "ISI"
avec les références aux pages de ces 2 volumes.

Une solution est offerte pour chacun des projets présentés dans ces 2 volumes. Les programmes contenus dans ces volumes utilisent uniquement les [commandes], les [fonctions] et les [instructions] BASIC ayant été apprises dans les pages qui précèdaient ces programmes...sauf quelques exceptions qui donneront le goût "d'attaquer" le chapitre suivant!

Après chaque chapitre, les élèves feront une liste de projets qu'ils aimeraient réaliser...Ils en discuteront entre eux et avec le professeur...Il est normal que certains projets intéressants soient retardés ou construits par modules en suivant les étapes d'apprentissage de la programmation. D'autres projets seront "mis sur les tablettes"...L'important est de concrétiser un ou des projets imaginés par les élèves...De l'imagination, ils en ont "à revendre"!

Par exemple les élèves pourraient réaliser un projet (statique) à la fin du chapitre un et conserver ce dessin sur cassette ou sur disque. Reprendre ce projet pendant l'étude du chapitre deux afin de l'animer...

Une remarque pour compléter ces quelques lignes d'introduction. Le Ministère de l'Education prévoit dans son programme:
a) Connaitre le fonctionnement matériel et l'histoire de l'ordinateur
b) Evaluer la place et le rôle de l'ordinateur dans la société
c) Développer des attitudes critiques vis-à-vis du traitement automatique de l'information

Même si cette partie du programme est traitée dans le chapitre 5 du volume 2, ces notions ne devront en aucun cas être donnnées à la fin de l'année scolaire dans un cours "ex cathédra" et in extremis!...

Ces notions seront planifiées et offertes aux moments opportuns au cours de l'année...et par petites doses de 5 ou 10 minutes (maximum).

Dépôt légal: novembre 1983
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Les photographies sont de Jean-Marc Mailhot, s.c.

Nous remercions par avance les personnes qui voudront bien nous adresser leurs commentaires, corrections, suggestions, remarques... en vue de la prochaine édition:

Laurian Picard, Collège d'Arthabaska, 905 boul. Bois-Francs sud, Arthabaska, Québec. G6P 5W1.

TABLE DES MATIERES

UN DEPART

La publicité n'exagère pas lorsqu'elle affirme: "Rien ne peut battre mon COMMODORE 64". Rien!...c'est-à-dire aucun autre ordinateur actuellement sur le marché ne peut offrir à un prix comparable les étonnantes possibilités du C64

Tu as fait un bon investissement et tu as raison d'en être fier car le C64 est un ordinateur "bien racé" qui profite d'une percée technologique importante avec son synthétiseur de musique. Son microprocesseur est le 6510; en fait c'est le célèbre 6502 auquel on a ajouté des "BUS".

Le C64 est plus difficile à programmer que le VIC-20 mais tu constateras rapidement que ce livre rend simple ce qui paraissait compliqué. Tu apprendras à construire des "SPRITES" dès les premiers chapitres afin de profiter de cet outil merveilleux pour l'animation de tes programmes.

Le C64 n'est pas seulement musicien et décorateur, il peut calculer tes problèmes de math (au primaire ou à l'université), établir des statistiques, tracer des courbes trigonométriques...organiser ton budget... Il est très efficace dans la correction des programmes, le traitement d'un texte ou de tout un volume comme celui que j'écris présentement à l'aide du logiciel "Paperclip".

Son clavier possède 64 touches dont une quarantaine offrent un choix de 3 caractères et même 4 caractères pour chacune des touches des chiffres 1 à 8... On peut dire que le C64 a "du caractère" et c'est sa "force de frappe" face à ses concurrents!

Quand tu mets le C64 et la TV (ou moniteur) sous tension, l'écran affiche:

```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

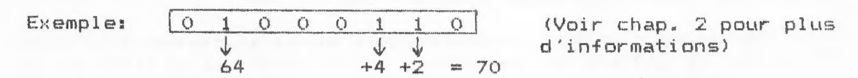
READY
█
```

La première ligne indique que le C64 offre la version 2 de BASIC.

La deuxième ligne te dit qu'il y a 64K ou 64 fois 1024 (65535) mémoires dans le C64 pour le 'SYSTEME' et pour les mémoires vives (anglais:RAM) et que 38911 octets (anglais: bytes) sont libres (FREE) c'est-à-dire qu'ils sont disponibles pour écrire tes programmes.

READY t'informe que le C64 <u>est prêt</u> à recevoir tes instructions. Le <u>curseur</u> est le petit rectangle qui clignote.

Un <u>octet</u> contient <u>8 bits</u> ou 8 unités d'information. Chaque bit peut prendre deux états: soit l'état 0 ou l'état 1.



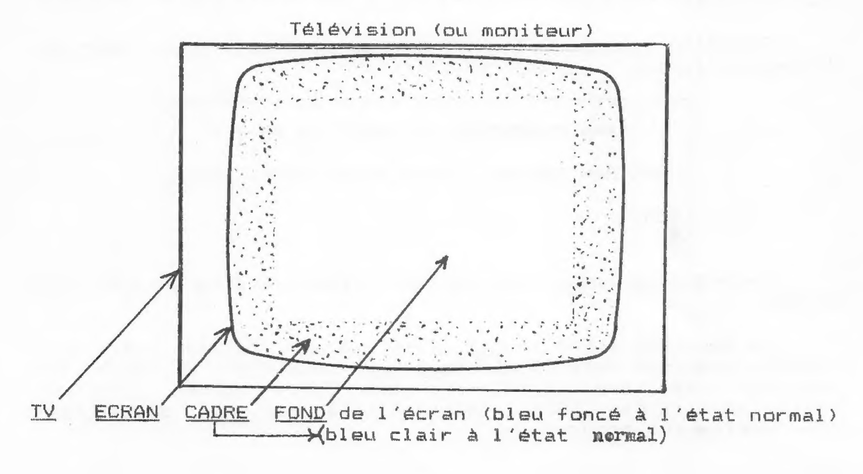
Cet octet est le profil binaire de la <u>lettre F</u> car le code de ce caractère F est 70 (voir tableau page 76).

Chaque caractère, même le caractère espace, demande 1 octet. Tu peux donc entrer environ 38900 caractères dans la mémoire de ton C64... soit à peu près l'équivalent d'une douzaine de pages de ce volume.

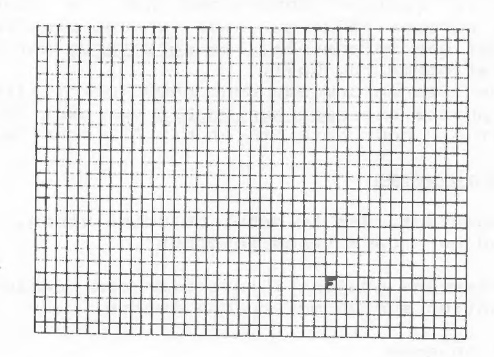
Ces 38911 octets forment une partie de la <u>ME</u>moire <u>V</u>ive (MEV) c'est-à-dire des mémoires dans lesquelles tu peux <u>lire ou écrire</u>. (anglais: RAM)

Le 'SYSTEME' (Kernal) du C64 est formé de MEmoires Mortes (MEM), en anglais (ROM), qui contiennent les programmes invariables que le constructeur a conçus pour permettre à l'utilisateur un échange interactif avec son C64.

N'oublie pas que les ordinateurs ne comprennent que le langage machine ou binaire. L'interpréteur BASIC est un de ces programmes utilitaires (en mémoire morte) qui transforme en binaire les caractères que tu tapes sur le clavier.



Le FOND de l'écran (partie active)

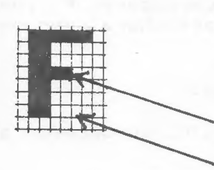


Le FOND de l'écran (i.e. la partie active) est divisé en 25 rangées (0-24) et en 40 colonnes (0-39). Tu te rappelleras que les numéros des rangées et des colonnes commencent par 0 et non par 1. Ce qui fait un total de 1000 petites cases ayant les dimensions du curseur. La lettre F a été tapée sur la rangée 20, colonne 27.

EXERCICE 1 Appuie sur la touche CRSR(bas) et sur la touche CRSR(droite) pour amener le curseur à la rangée 20, colonne 27 et tape le caractère F.

L'information est immédiatement codée en binaire par l'interpréteur et le C64 affiche la lettre F selon les coordonnées (20,27). Autrement dit le C64 allume les points qui tracent le dessin de la lettre F à la case (20,27).

Agrandissement (8 fois)
de la case (20,27) ci-dessus



Chacune des 1000 cases du FOND de l'écran est formée de 64 autres cases aussi petites que la pointe d'un crayon. C'est pourquoi on dit que chaque case est formée de 64 points (anglais: dots).

Le caractère F: couleur bleu clair (20 points sont allumés)

Fond de l'écran de la case (20,27): couleur bleu foncé (44 points sont éteints)

On dit qu'un point est éteint quand il est de la couleur qui a été choisie pour le FOND de l'écran (Ici c'est le bleu foncé, état normal du C64).

On dit que les points des caractères sont allumés quand ils sont de la couleur déterminée par la zone mémoire commençant à l'adresse 55296 que nous verrons plus loin. Quand la couleur n'est pas déterminée, les caractères ont la couleur bleu clair (état normal du C64).

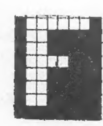
Le FOND de l'écran possède donc 64000 points (1000 X 64). Chacun de ces points est codé sur un bit qui peut être allumé ou éteint: bit à 0 pour "éteint" et bit à 1 pour "allumé".

Les caractères **inversés**:

Il est important, dès le début de notre étude, de savoir ce qu'on entend par "caractères inversés".

Nous utiliserons souvent l'inversion des caractères pour attirer l'attention sur un mot ou une lettre.

Le caractère F inversé



Dans la case où se trouve un caractère inversé, ce sont les points du caractère qui sont éteints (couleur bleu foncé du fond de l'écran).

Tous les autres points sont allumés (couleur bleu clair dans l'état normal du C64).

EXERCICE 2:

a) Tiens la touche SHIFT enfoncée et appuie sur les touches CRSR(haut) et CRSR(gauche) pour amener le curseur sur la ligne 18, colonne 27.

b) Appuie simultanément sur la touche CTRL et sur la touche RVSon (chiffre 9). N'oublie pas que la touche CTRL doit être enfoncée pendant que tu appuies sur RVSon.

c) Tape la lettre F et tu obtiendras le caractère F inversé. Appuie sur d'autres touches et l'écran affichera encore des caractères inversés.

QUESTION 1 Comment revenir à l'état normal?

Réponse: Pendant que tu tiens la touche CTRL enfoncée, appuie sur RVSoff.

QUESTION 2 Combien y a-t-il de bits allumés dans le rectangle qui clignote (curseur) lorsqu'il apparaît sur l'écran?

Réponse: Tous les 64 bits sont allumés.

QUESTION 3 Le curseur apparaît sur l'écran pendant 1/3 de seconde et devient invisible pendant 1/3 de seconde. Combien de bits demeurent allumés quand le curseur devient invisible (bleu

sur fond bleu)?

Réponse: Tous les bits sont à 0 (éteints).

EXERCICE 3 Inversion du caractère espace:

Appuie quelques fois sur la barre d'espacement (la longue touche rectangulaire en bas du clavier). Tu viens d'afficher quelques espaces sur l'écran mais ces espaces sont invisibles parce que tous les bits du caractère espace sont à 0 (couleur du fond de l'écran: bleu sur bleu).

Maintenant tu vas allumer le caractère espace. Tiens la touche CTRL enfoncée pendant que tu appuies sur RVSon.

C'est fait! Le caractère espace est maintenant bleu clair (couleur des caractères dans l'état normal du C64).

Appuie la barre d'espaces et l'écran affichera une longue bande bleu clair. On dit que le caractère espace a été inversé. On utilise régulièrement 'l'espace inversé' pour dessiner des silhouettes.

Remets le C64 à l'état normal: RVSoff (Voir la question 1).

EXERCICE 4 Obtention des couleurs:

Sur la partie verticale des touches 1 à 8, tu peux lire l'abréviation anglaise de 8 couleurs:

1- BLK = noir	2- WHT = blanc	3- RED = rouge
4- CYN = turquoise	5- PUR = pourpre	6- GRN = vert
7- BLU = bleu	8- YEL = jaune	

Dessine la drapeau français (bleu blanc rouge) en utilisant le caractère espace inversé. Note: Le drapeau se trouvera suspendu en position verticale (hampe horizontale et en haut de l'écran

SOLUTION

a) Il faut d'abord "effacer" l'écran. Appuie simultanément sur la touche SHIFT et sur la touche CLEAR HOME. L'écran se vide et seul le curseur se trouve sur l'écran, en haut à gauche. On dit que le curseur est à l'origine, aux coordonnées (0,0).

b) Comme le fond de l'écran est déjà bleu foncé, tape 5 fois sur CRSR(bas) pour descendre le curseur de 5 rangées

c) Appuie simultanément CTRL et RVSon pour obtenir des caractères inversés et simultanément CTRL et chriffre 2 pour obtenir le 'blanc'.

d) Appuie la barre d'espaces et laisse dérouler le trait blanc

sur 5 lignes d'écran. Note: Quand tu veux effacer derrière le curseur, tu appuies la touche INST DEL.

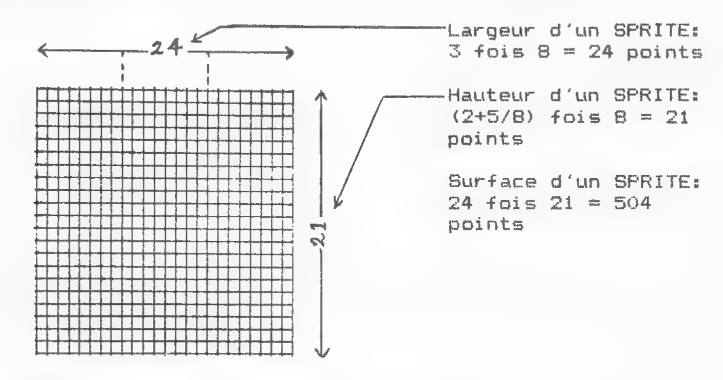
e) Le C64 est toujours à l'état RVSon depuis l'étape c). Appuie simultanément sur la touche CTRL et la touche RED (chiffre 3). Appuie ensuite la barre d'espaces jusqu'à ce que tu obtiennes 5 lignes de couleur rouge.

UN SPRITE (définition)

Un Sprite est un dessin que tu construis toi-même et que tu introduis dans la mémoire du C64 pour l'afficher ensuite n'importe où sur l'écran sans que les caractères produits par les touches du clavier soient effacés

Les dimensions d'un SPRITE

Tu as appris, à la page 6, que le fond de l'écran est formé de 64000 petits points (anglais:dots)
Largeur de l'écran: 40 fois 8 = 320 points
Hauteur de l'écran: 25 fois 8 = 200 points
Surface de l'écran: 320 multiplié par 200 = 64000 points

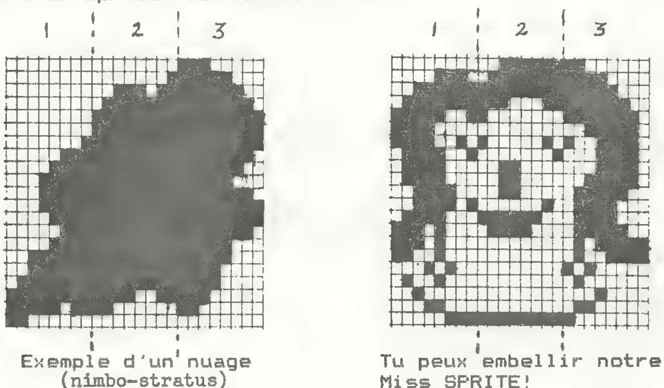


JETTE UN REGARD sur la page 21 et tu auras une idée des dimensions réelles d'un Sprite en comparaison avec les dimensions réelles du fond de l'écran (TV 36 cm ou 14 pouces)

L'ORDINATEUR pourra doubler les dimensions d'un sprite si tu lui donnes 2 petites instructions (chapitre 2 p.55)

EXERCICE 5 Utilise le dessin quadrillé ci-dessus (24X21) pour en faire plusieurs calques et, à l'aide de ces calques, dessine un nuage et une "Miss SPRITE"

Tu as une solution ci-dessous...Les dimensions des 2 dessins sont environ 3 fois plus grandes que les dimensions normales d'un sprite (sur écran 36 cm).



Exemple d'un nuage (nimbo-stratus)

Tu peux embellir notre Miss SPRITE!

AU CHAPITRE 2 tu apprendras comment introduire ces sprites dans la mémoire de ton C64 pour ensuite les afficher et les déplacer sur l'écran. En attendant, mets tes dessins en lieu sûr!...

LES DEUX MODES DE FONCTIONNEMENT DU C64

1- LE MODE DIRECT

Dans le mode direct, les instructions que tu donnes au C64 sont exécutées aussitôt que tu tapes la touche RETURN.

Avant de faire l'exercice 5, je te signale que les parenthèses s'obtiennent en appuyant simultanément sur la touche SHIFT et sur les chiffres 8 ou 9.

EXERCICE 6 Ecris en mode direct:(maximum 2 lignes sur écran);

```
POKE 53281,1:PRINT CHR$(147);:FOR I=1 TO 200:PRINT CHR$(18)
CHR$(32);:NEXT I
```

Le curseur se trouve positionné après NEXT I. Ne le déplace pas afin d'être sûr qu'il est dans la "ligne BASIC".

Tape la touche RETURN et l'écran affichera la bande bleue du drapeau de la France sur un FOND blanc.

Si le C64 te signale une erreur, ne t'en fais pas! Ecris

de nouveau la ligne en t'assurant que tu tapes les bons caractères. Ne pas confondre le chiffre 0 qui est traversé par un trait oblique (sur l'écran) et la lettre O!

EXERCICE 7 Ecris en mode direct:

PRINT 200-50+25 Tape RETURN. L'écran affiche 175.

Nous aurons l'occasion plus loin d'expérimenter ce mode direct dans des calculs plus complexes tout comme avec une calculatrice.

Le mode direct est utile pendant l'élaboration d'un programme... et chaque fois qu'on veut obtenir un résultat immédiat qui ne doit pas être conservé.

Il faut te rappeler qu'en mode direct, tes instructions ne peuvent pas être récupérées lorsqu'elles disparaissent de l'écran.

EXERCICE 8 Ecris en mode direct: (sur 2 lignes d'écran)

```
FOR I=1 TO16:POKE 53280,I:FOR J=1 TO16:POKE 53281,J:
FORT=1TO500:NEXT:NEXT:NEXT
```

RETURN...Tu obtiens les couleurs du cadre et de l'écran.

2- LE MODE PROGRAMME (ou différé)

Le mode programmé possède toujours un numéro au début de la ligne BASIC. De plus, même si ton programme n'est plus sur l'écran, tu peux l'afficher de nouveau au moyen de la commande LIST.

N'oublie pas de taper RETURN après chaque ligne BASIC.

EXERCICE 9 Ecris le programme suivant:

```
10 PRINTCHR$(147);:POKE 53281,15
20 FORI=1TO200:PRINTCHR$(18)CHR$(31)CHR$(32);:NEXT
30 FORJ=1TO200:PRINTCHR$(5)CHR$(32);:NEXT
40 FORK=1TO200:PRINTCHR$(28)CHR$(32);:NEXT
```

Au lieu des numéros 10, 20, 30,...tu peux écrire 1, 2, 3,... Cependant, il est conseillé de laisser des numéros libres afin de pouvoir insérer d'autres instructions pendant la construction d'un programme plus élaboré.

Ton petit programme est dans la mémoire du C64. Ecris RUN et appuie RETURN. L'exécution du programme commence.

A la fin, l'écran affiche le drapeau de la France (position verticale), le même résultat que tu as obtenu en mode direct à l'exercice 4.

EXERCICE 10 Au-dessous de READY, là où le curseur clignote, ajoute ces lignes 15 et 16 à l'exercice 9 pour dessiner une hampe et 4 attaches...:

```
15 FORL=1TO40:PRINTCHR$(18)CHR$(144)CHR$(32);:NEXT:PRINTCHR$(146);
16 FORM=1TO4:PRINT "|",:NEXT
```

Si tu veux afficher la liste de ton programme sur l'écran: Ecris LIST et tape RETURN. Pour faire exécuter de nouveau ton programme, écris RUN et tape RETURN.

Avant de commencer le chapitre UN, fais cet exercice:

EXERCICE 11 Ecris ce programme et relis chacune des lignes pour vérifier s'il ne manque pas un caractère, une parenthèse, un signe... N'oublie pas de tourner le bouton volume de la TV!

```
10 PRINT CHR$(147): X=17: S=54272: POKE 53281,1: POKE S+24,15
20 FOR I=1 TO 8: READ HF,BF: H(I)=HF: B(I)=BF: NEXT I
30 POKE S+5,88: POKE S+6,89
40 E=INT(RND(1)*1000)
45 C=INT(RND(1)*16)
50 N=INT(RND(1)*8+1)
55 F=INT(RND(1)*18+16)
60 IF F=33 OR F=17 THEN X=F
65 POKE S+1,H(N): POKE S,B(N)
70 POKE 1024+E,81+C: POKE 55296+E,C
75 POKE S+4,X: FOR T=1 TO 250: NEXT T
80 POKE S+4,X-1: FOR T=1 TO 50: NEXT T
85 GOTO 40
90 DATA 17,37,19,63,21,154,22,227
95 DATA 25,177,28,214,32,94,34,75
```

pour arrêter le programme, appuie simultanément la touche RUN/STOP et la touche RESTORE

Dans ce programme, les valeurs des variables sont choisies au hasard (RND): 1- le caractère (un des numéros 81 à 96 page 76) 2- sa position sur l'écran (0 à 999) 3- sa couleur (0 à 15) 4- la note musicale (Do-4 à DO-5) 5- forme de l'onde (17 ou 33).

LES INSTRUCTIONS BASIC QUE TU AS utilisées DANS CES PAGES D'INTRODUCTION seront expliquées DANS LES CHAPITRES SUIVANTS.

Avec de la patience, de la ténacité et beaucoup d'exercices, tu pourras réaliser toi-même d'intéressants programmes quand tu auras terminé l'étude pratique de ce premier volume.

BON COURAGE!

CHAPITRE I

ACTIVITES DIRIGEES: Série No 1

BUTS: Cette première série d'activités a pour but de te familiariser avec:

1- les instructions PRINT et GOTO

2- les commandes LOAD, NEW, RUN, SAVE, VERIFY, LIST

3- les deux touches majuscules [SHIFT]

4- la touche Commodore [C=] (emblème de Commodore)

5- les 2 touches du curseur [⇧ CRSR] avec SHIFT [⇦ CRSR] avec SHIFT

et [CRSR ⇩] sans SHIFT [CRSR ⇨] sans SHIFT

6- la barre d'espacement [ ] En appuyant sur cette barre tu mets un blanc (un espace) sur l'écran... Donc tu effaces vers la droite

7- la touche qui permet d'effacer vers la gauche [INST DEL] sans SHIFT

8- les caractères graphiques

9- la touche qui emmagasine en mémoire ou exécute une ou des instructions [RETURN]

10- enfin, sur les 11 CARACTRES DE COMMANDE D'AFFICHAGE, tu en apprendras six: voir page 35 numéros 1, 7, 8, 9, 10, 11.

PRINT

Il n'y a pas d'instruction en BASIC qui présente autant de variétés d'applications que l'instruction PRINT. Il faudra que tu lui apportes une attention spéciale et que tu fasses beaucoup d'exercices pour profiter de toutes les possibilités que cette instruction peut t'offrir.

EXERCICE 1 Ecris en mode direct: (pas de numéro de ligne)

PRINT 100*4/2+25 Tape RETURN
* est le signe de la multiplication
/ est le signe de la division

L'écran affiche 225. Nous verrons plus loin les autres symboles mathématiques.

Le mot anglais PRINT (imprimer) devrait en principe imprimer les résultats sur une feuille au moyen d'une imprimante. Et le mot DISPLAY (afficher) devrait servir d'instruction pour afficher les résultats sur un écran.. Avec l'avènement des moniteurs et TV, on a préféré conserver, pour des raisons historiques, le même mot PRINT pour dire "imprimer" et "afficher". Dans ce volume nous emploierons indistinctement imprimer ou afficher.

EXERCICE 2 Ecris en mode direct:

PRINT"BON COURAGE!" et appuie RETURN

L'écran affiche tous les caractères placés entre les guillemets.

1- Instruction PRINT **avec ou sans** guillemets

Quand le C64 rencontre une instruction PRINT, il se trouve en présence de 2 choix. Il se demande: est-ce qu'on veut que **j'imprime des caractères** où si on veut que **je fasse des calculs**? C'est toi qui vas indiquer après PRINT le choix que l'ordinateur fera.

a) Si tu mets des caractères entre guillemets, le C64 affichera intégralement l'information contenue dans ces guillemets: les chiffres, les espaces, les lettres, etc...

b) S'il n'y a pas de guillemet après PRINT, le C64 saura qu'il doit faire, soit un calcul avec les chiffres, les fonctions ou les lettres qu'il considérera comme des variables numériques, ou soit imprimer une chaîne de caractères si la variable possède le signe dollar (Nous parlerons plus tard de ces variables avec $).

EXERCICE 3 Ecris en mode direct:

PRINT "15 MAI"   RETURN

L'écran affiche 15 MAI. Donc l'ordinateur imprime exactement ce qu'il découvre entre les parenthèses.

EXERCICE 4 Ecris en mode direct

PRINT 15 MAI   RETURN

L'écran affiche 15 0. Comme il n'y a pas de guillemet, le C64 veut exécuter un calcul et il t'indique que la variable MAI vaut 0. Il faut noter qu'il y a deux espaces entre 15 et 0: un espace après 15 et un espace avant 0.

L'ordinateur met automatiquement un espace avant et après les nombres. Si le nombre est négatif, l'espace avant sera remplacé par le signe -.

EXERCICE 5 Ecris en mode programmé (il faut un numéro de ligne)

```
10 A = 10 * 5
20 PRINT"VALEUR DE A ="A
```

L'écran affiche: VALEUR DE A = 50 Pour faire exécuter un programme il faut d'abord écrire RUN et ensuite appuyer la touche RETURN pour que ta commande RUN puisse passer dans la mémoire du C64.

Dans l'instruction 10 tu demandes ceci au C64: multiplie 10 par 5 et mets le résultat dans la variable A. A l'instruction 20, tu lui demandes: affiche les caractères placés entre les guillemets et imprime la valeur contenue dans la variable A.

2- Abréviation de l'instruction PRINT

Il est temps que tu apprennes que PRINT peut être remplacé par un symbole très court: le [?]. Tu vas sauver du temps en tapant un seul caractère au lieu de 5.

Ecris de nouveau ton programme avec les mêmes numéros de ligne et les anciennes instructions 10 et 20 s'effaceront automatiquement de la mémoire du C64 pour être remplacées par les nouvelles.

EXERCICE 6 Ecris en mode programmé

```
10 A = 10 * 5
20 ? "VALEUR DE A =" A              (RUN et RETURN)
```

Tu obtiens les mêmes résultats... Maintenant, vérifie si le C64 a bien inscrit PRINT dans sa mémoire: Appuie simultanément SHIFT et CLR HOME pour effacer l'écran. Ecris

LIST et appuie RETURN. Le [?] est bien remplacé par PRINT!

Pour effacer l'ancien programme de la mémoire tu écris NEW et appuie RETURN.

Tu peux vérifier ensuite que la mémoire BASIC est vide; écris: LIST et appuie RETURN. Il n'y a aucune instruction sur l'écran entre LIST et READY. Ecris RUN et RETURN... Le C64 affiche seulement le mot READY.

QUESTION 1 Les instructions 10 et 20 sont demeurées en haut de l'écran, pourquoi?

Réponse: NEW n'efface pas l'écran. Profites-en pour faire une expérience:

Remonte le curseur sur la ligne 10 en appuyant sur SHIFT et la touche CRSR(haut). Tape ensuite RETURN.

Le curseur s'est positionné automatiquement sur la ligne 20. Tape encore sur RETURN. Ton programme est de nouveau en mémoire car un des rôles de la touche RETURN est d'emmagasiner les instructions dans la mémoire du C64. Ecris RUN et RETURN et tu verras tes résultats de nouveau sur l'écran.

3- **Rôle des caractères [,] et [;] dans les instructions PRINT**

Les caractères virgule [,] et point virgule [;] sont utilisés comme **indicateurs** dans une instruction PRINT.

a) La **virgule indique** à l'ordinateur que tu veux faire imprimer tes résultats sur 4 sections de l'écran: colonne 0, colonne 10, colonne 20 et colonne 30 (note: automatiquement colonnes 1,11,21 et 31 pour les chiffres afin d'avoir une case libre pour un éventuel signe -).

EXERCICE 7 Ecris en mode programmé

```
10 PRINT -1,4,-6,8
20 PRINT "MOINS UN","QUATRE","MOINS SIX","HUIT"
```

L'écran affiche:

```
-1        4         -6        8
MOINS UN  QUATRE    MOINS SIX HUIT
```

Tu peux mettre un maximum de 9 caractères par section...Si tu dépasses cette valeur, le C64 n'imprimera rien sur la section suivante

EXERCICE 8 Ecris en mode programmé

```
10 PRINT "JE","DEMEURE","AMICALEMENT","VOTRE"
```

Le mot VOTRE n'a pas été imprimé sur la 4ième section parce que AMICALEMENT possède plus de 9 caractères.

b) Dans une instruction PRINT, **le point virgule indique** à l'ordinateur qu'il doit imprimer **la valeur suivante immédiatement derrière la valeur précédente.** Autrement dit, les valeurs s'impriment les unes à la suite des autres.

Remplace les 3 virgules de l'exercice 8 par des points virgules.

L'écran affichera: JEDEMEUREAMICALEMENTVOTRE

Il faut donc introduire un caractère espace à la fin des trois premiers mots, avant de fermer les guillemets

EXERCICE 9-A Ecris en mode programmé

```
10 PRINT "JE ";"DEMEURE ";"AMICALEMENT ";"VOTRE"
```

EXERCICE 9-B Maintenant que tu connais le rôle de la [,] et du [;] tu peux écrire simplement:

```
10 PRINT "JE DEMEURE"
20 PRINT
30 PRINT "AMICALEMENT VOTRE"
```

Une instruction qui contient seulement le mot PRINT imprimera une ligne vide (passera une ligne). C'est souvent utile pour aérer un texte.

Avant de passer à l'instruction GOTO, fais ce petit dessin à l'aide des caractères graphiques qui se trouvent sur la partie verticale des touches du clavier (note: le C64 se chargera de fermer automatiquement les guillemets).

EXERCICE 10

```
5 PRINTCHR$(147)
10 PRINT"  ╱‾‾╲
20 PRINT" ╱    ┐
30 PRINT" └────■
40 PRINT"  O  O
```

> Avec PRINT la fonction CHR$(147) permet d'effacer l'écran. Voir chapitre III.

Explications

Ligne 10: après avoir ouvert les guillemets, tu mets 2 espaces, tu appuies SHIFT et N, 2 fois C= et T, SHIFT et M.
Ligne 20: 1 espace, SHIFT et N, 4 espaces, SHIFT et P.
Ligne 30: 1 espace, SHIFT et L, 4 fois Commodore et @, Commodore et + .
Ligne 40: 2 espaces, SHIFT et W, 2 espaces, SHIFT et W.

GOTO (ou GO TO)

L'instruction GOTO est une instruction de branchement inconditionnel. Dans ce cas l'exécution du programme est renvoyée au numéro de ligne qui est spécifié après GOTO.

EXERCICE 11

```
10 PRINT"*";
20 GOTO 10
```

> Pour arrêter le programme appuie la touche RUN/STOP

Le caractère **;** fait imprimer les étoiles les unes à la suite des autres. L'instruction GOTO 10 fait recommencer indéfiniment l'exécution du programme.

Je te conseille d'organiser tes programmes de façon à limiter le plus possible l'emploi de l'instruction GOTO... Ainsi il sera plus facile de faire une lecture séquentielle de tes programmes.

EXERCICE 12

```
5 PRINTCHR$(147)
10 PRINT"/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\"
20 PRINT"\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/"
30 GOTO 30
```

> Tu obtiens les traits obliques en appuyant SHIFT et les touches N ou M.

Ici GOTO retourne au numéro de sa propre instruction. Le programme "tourne en rond" dans la ligne 30 et empêche ainsi le C64 de faire apparaître le mot: **READY** et le curseur

Quand un programmeur construit un dessin sur l'écran, il utilise cette technique pour empêcher le mot READY d'en détruire une partie. Il efface ensuite cette instruction car il est interdit, en principe, de laisser une boucle infinie dans un programme!

Nous utiliserons cette technique dans nos projets de ce chapitre I. Pour revoir READY et le curseur, appuie la touche RUN/STOP.

Les 11 Caractères de COMMANDE D'AFFICHAGE

Les 11 caractères de commande d'affichage sont illustrés et expliqués dans l'important tableau de la page 35. Ils sont numérotés et ce sont ces numéros de référence que nous allons utiliser dans ce volume.

Il est important que tu deviennes un "champion" dans l'usage de ces caractères car ils possèdent d'étonnantes possibilités d'édition. Nous allons faire quelques exercices avec **les numéros 1, 7, 8** que tu auras à utiliser dans tes projets de ce premier chapitre.

EXERCICE 13-A Ecris en mode direct:

PRINT"♥"

Pour obtenir ce caractère tu tiens la touche **SHIFT** enfoncée et tu appuies sur **CLR/HOME**(voir no 1 page 35).

Tape RETURN. Tu constates que ce caractère de commande d'affichage efface l'écran. Son symbole est un coeur en contraste inversé.

EXERCICE 13-B Ecris en mode direct:

PRINT"RVACANCES

Pour obtenir ce caractère tu tiens la touche **CTRL** enfoncée et tu appuies sur **RVSon** (voir no 7 page 35).

Tape RETURN. Le mot VACANCES apparaît en caractères inversés. Le symbole de ce caractère de commande d'affichage est un R en contraste inversé. Une nouvelle instruction PRINT remet automatiquement le C64 à l'état normal (RVSoff).

EXERCICE 13-C Ecris en mode direct:

PRINT"RVIVE■ LES RVACANCES

Le caractère placé après VIVE s'obtient en appuyant à la fois sur **CTRL et RVSoff** (voir no 8 page 35).

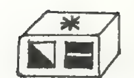
Tape RETURN. Le mot VIVE s'imprime en contraste inversé, le mot LES revient à l'état normal parce qu'il est précédé par le caractère de commande d'affichage no 8 et VACANCES s'imprime en contraste inversé.

Le symbole de ce caractère no 8 est un rectangle en contraste inversé qui possède une ligne éteinte (ici une ligne blanche) sur l'avant dernière rangée.

Il faut t'exercer à reconnaître rapidement ces symboles de commande d'affichage dans la liste d'un programme. Ces symboles sont plus difficiles à lire si tu utilises l'imprimante 1525 qui imprime ses caractères dans une matrice de 42 points (6 X 7) tandis que sur l'écran tu sais déjà que la matrice d'un caractère possède 64 points (8 X 8).

Voici quelques précisions concernant les caractères graphiques avant de faire l'exercice 14. Tu as remarqué que plusieurs touches possèdent 2 caractères graphiques sur la partie verticale avant de ces touches.

Prenons l'exemple de la touche étoile:



Le caractère de gauche est un triangle rectangle blanc placé dans une petite case qui <u>symbolise</u> une <u>case de</u> l'écran. C'est ce triangle qui s'imprimera dans l'état normal du C64.
Pour obtenir le triangle opposé ——→
il faut mettre le C64 en RVSon.

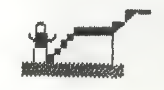
Il en est ainsi pour le caractère de droite. A l'état normal du C64, l'écran affichera la ligne horizontale au milieu de la case et tous les autres points seront éteints. Avec RVSon tu auras le caractère inversé: la ligne horizontale du milieu sera éteinte (couleur du fond de l'écran) et tous les autres points seront allumés (état normal: bleu clair.

<u>Pour afficher</u> le caractère graphique <u>de gauche</u>, il faut tenir la touche <u>C</u>ommodore (C) enfoncée...

<u>Pour afficher</u> le caractère graphique <u>de droite</u>, il faut tenir la touche SHIFT enfoncée...

<u>EXERCICE 14</u> Fais l'exercice suivant et (n'oublie pas de taper RETURN après chaque ligne

```
10 PRINT"          ◤▔▔"
20 PRINT" O   ▄▄▄▄▄◢"
30 PRINT"╲▨ ▀◢▚▄▄    |"
40 PRINT"||◢▄▄     |"
50 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒"
```



Tu as toutes les connaissances nécessaires pour construire ce petit dessin...

<u>Les deux touches CRSR</u>

Ce sont deux touches importantes qui permettent de déplacer le <u>CURSEUR</u> dans les 4 orientations: EST, OUEST, NORD, SUD. Autrement dit: à droite, à gauche, en haut, en bas.

Il y a des flèches sur les touches CRSR pour t'indiquer l'orientation. Le déplacement du curseur vers le bas et vers la droite se font simplement en appuyant sur la touche CRSR (sans SHIFT).

Les déplacements vers le haut et vers la gauche exigent que tu appuies <u>d'abord sur SHIFT</u>.

<u>EXERCICE 15</u>

Examine d'abord la page 21. Elle a les dimensions du fond de l'écran d'une TV 36 cm (14 pouces). Donc chacune des 1000 cases a les <u>dimensions réelles</u> du curseur. Tu pourras faire plusieurs calques de cette page et les utiliser pour préparer tes dessins.

Amène le curseur vers le centre de l'écran et fais-le cheminer selon le tracé indiqué à la page suivante.

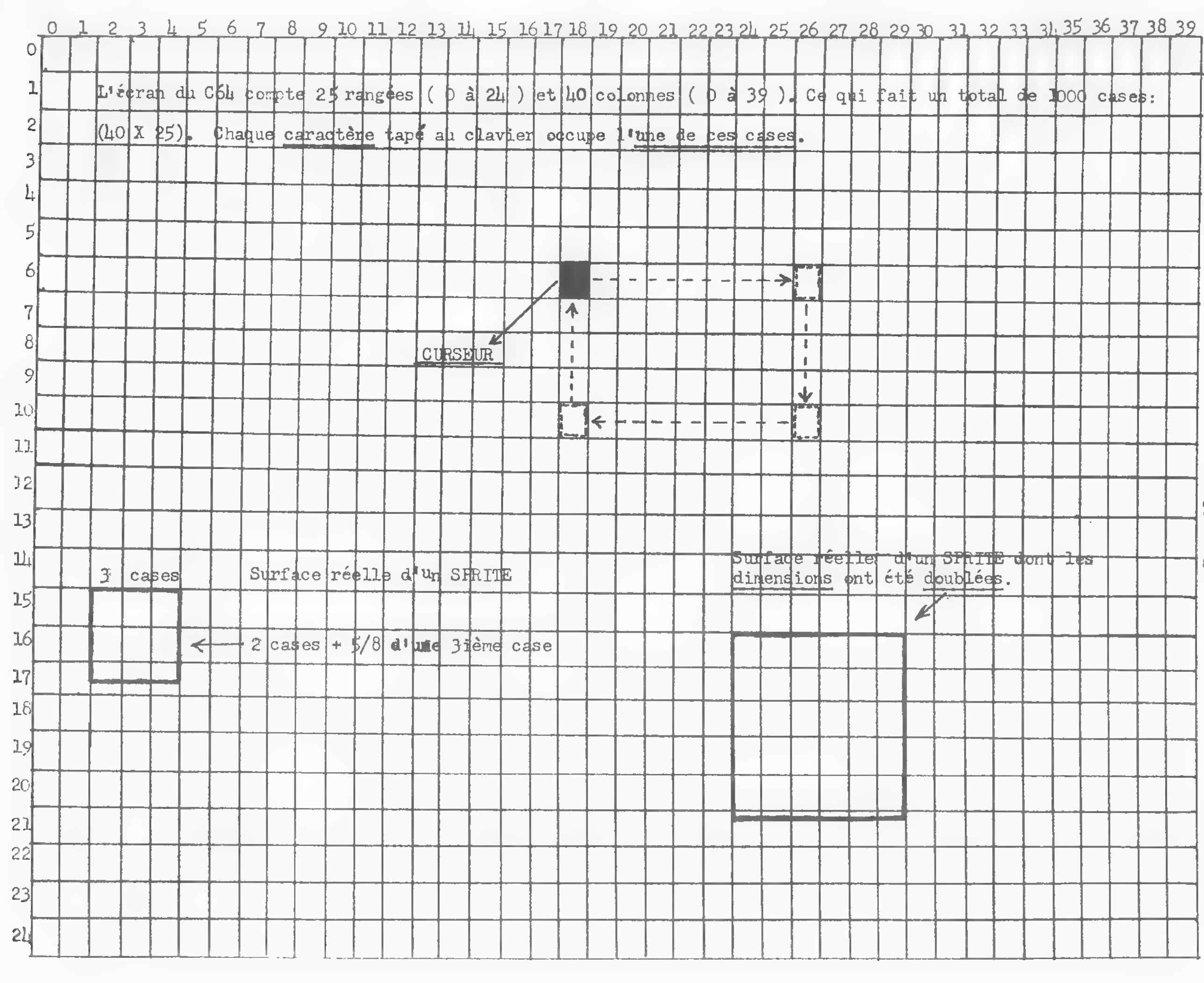
Repars d'un autre endroit de l'écran et refais plusieurs fois cet exercice... Il faut que tu maîtrises "les contrôles curseurs" aussi aisément que le bras des vitesses d'une auto!

Imprime des étoiles pendant que tu déplaces le curseur sur un chemin rectangulaire.

Enfin, utilise la touche INST DEL sans SHIFT pour effacer les étoiles qui forment les contours de ton triangle.



Photo du sous-programme "Fleur de Lis", page 158


L'écran du C64 compte 25 rangées (0 à 24) et 40 colonnes (0 à 39). Ce qui fait un total de 1000 cases:
(40 X 25). Chaque caractère tapé au clavier occupe l'une de ces cases.
CURSEUR
3 cases
Surface réelle d'un SPRITE
2 cases + 5/8 d'une 3ième case
Surface réelle d'un SPRITE dont les dimensions ont été doublées.

PROJET A-1

a) Construis une maison avec un paratonnerre
-Indique les charges négatives et une bonne mise à la terre

-Installe une lampe sur le coin gauche de la maison

-Ajoute un soleil au-dessus de la maison. Au chapitre 2 tu introduira un SPRITE (nuage)

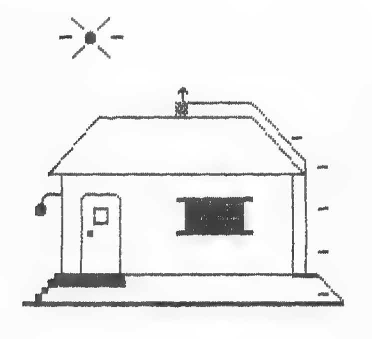
b) Entre ton dessin dans un programme par la METHODE des instructions PRINT.

c) Fais exécuter ton programme en écrivant RUN.

d) Sauve ton programme sur cassette ou disquette et vérifie s'il est bien implanté sur la bande magnétique.

A la page suivante, tu as un exemple de réalisation du projet A-1. Tu pourras faire mieux!

Voici ce que pourrait donner ton programme après avoir écrit RUN et appuyé sur RETURN.



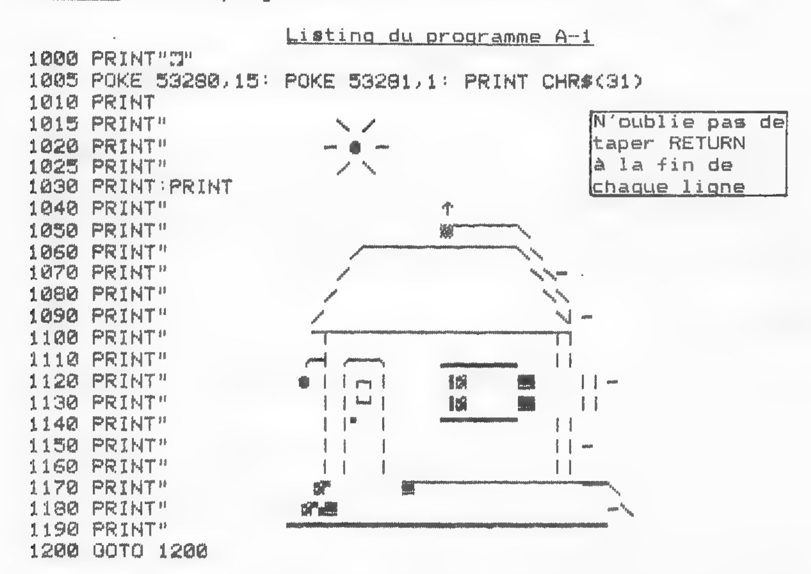
<u>Programme A-1</u>

sur l'imprimante VIC-1525

C'est plus joli sur l'écran

Si tu écris LIST et appuies sur RETURN tu obtiendras le <u>listing</u> de ton programme.

<u>Listing du programme A-1</u>



```
1000 PRINT"□"
1005 POKE 53280,15: POKE 53281,1: PRINT CHR$(31)
1010 PRINT
1015 PRINT"
1020 PRINT"
1025 PRINT"
1030 PRINT:PRINT
1040 PRINT"
1050 PRINT"
1060 PRINT"
1070 PRINT"
1080 PRINT"
1090 PRINT"
1100 PRINT"
1110 PRINT"
1120 PRINT"
1130 PRINT"
1140 PRINT"
1150 PRINT"
1160 PRINT"
1170 PRINT"
1180 PRINT"
1190 PRINT"
1200 GOTO 1200
```

N'oublie pas de taper RETURN à la fin de <u>chaque ligne</u>

Notes: 1- Chaque ligne du programme est en fait listée sur deux lignes, c'est pourquoi le dessin a cette forme allongée

2- Le document qui présente la liste complète des instructions d'un programme s'appelle: le <u>listing</u> (mot anglais francisé)

REALISATION

"C'est à ton tour de..."

Première étape

-Commence par effacer l'écran (voir no 1 page 35)

-Trace ton dessin selon tes goûts et ton talent d'artiste en utilisant les caractères graphiques.
Si tu as besoin d'aide pour ce premier dessin tu trouveras **une solution** détaillée à la page 28

-Tu peux effacer (corriger) ton dessin avec la barre d'espaces ou avec la touche

INST/**DEL** sans SHIFT

-Tu peux déplacer ton dessin ligne par ligne en plaçant le curseur à gauche du dessin et:

si tu veux que la ligne se déplace vers la droite, appuie sur

INST/DEL avec SHIFT

si tu veux que la ligne se déplace vers la gauche, appuie sur

INST/**DEL** sans SHIFT

-Tu peux aussi déplacer le dessin tout d'un bloc vers le haut si tu amènes le curseur sur la dernière ligne et que tu tapes RETURN. Note: Si tu veux envoyer le curseur au début de la ligne suivante pendant la construction d'un dessin ne possédant pas encore de No de ligne ni PRINT ni guillemet, tu dois appuyer SHIFT avant de taper RETURN pour éviter l'écriture de READY ou de SYNTAX ERROR.

Deuxième étape

Cette deuxième étape est une caractéristique puissante du C64 avec lequel tu travailles. Il suffit de taper, en tête de chaque ligne du dessin:

a) le numéro de l'instruction. Utilise des numéros de même longueur pour éviter les décalages dans le dessin.
b) le ? (abrégé de PRINT). Note: l'instruction PRINT est l'instruction la plus simple qui permet de faire des dessins sur l'écran.
c) ouvrir les guillemets. Le C64 se chargera de les fermer de façon fictive car ils n'ont été ouverts qu'une fois par ligne.
d) ne pas oublier de taper RETURN à chaque ligne.

Quand tu auras terminé cette étape, écris LIST et appuie sur RETURN. Tu verras défiler le listing de ton programme.

Si tu veux que les instructions se déplacent moins vite sur l'écran, appuie sur la touche CTRL. Tu reconnais ton dessin même s'il est listé sur deux lignes. Passe à l'étape trois et tu pourras demander au C64 d'afficher ton dessin sur l'écran.

Troisième étape

Ton dessin est maintenant dans la mémoire du C64. Pour savoir si ton programme fonctionne, écris: RUN et appuie sur RETURN.

RUN est la commande qui fera exécuter ton programme.

QUESTION 2 La partie du haut de mon dessin disparaît quand le curseur réapparaît en bas avec le mot READY. Quoi faire pour que mon dessin soit intact?

Réponse: Il suffit d'empêcher que le curseur apparaisse en ajoutant une dernière instruction. Si ta liste se termine avec le numéro 1180 (tu as un simple trait pour les fondations), alors tu ajoutes l'instruction suivante: 1190 GOTO 1190. Ainsi le C64 se trouvera emprisonné dans une boucle et il ne pourra pas te dire qu'il est prêt (READY) à faire autre chose.

QUESTION 3 Avec cette instruction no 1190 GOTO 1190 que j'ai ajoutée, mon programme ne s'arrête jamais. Quoi faire?

Réponse: Appuie sur STOP

QUESTION 4 Mon programme n'apparaît pas seul sur l'écran; ce que j'avais sur l'écran avant de faire RUN apparaît sur le haut de l'écran.

Réponse: Tu peux effacer l'écran en incluant une nouvelle instruction dans ton programme. Si ton dessin commence à l'instruction no 1010, écris l'instruction suivante:

```
1000 PRINT"♥"
```

↑ Caractère de "COMMANDE D'AFFICHAGE: No 1 page 35. Son rôle est de vider l'écran avant d'imprimer ton dessin

QUESTION 5 Comment changer les couleurs du cadre, de l'écran, et des caractères?

Réponse: Ajoute la ligne 1005 à ton programme. cette question sera étudiée au début du chapitre trois.

NOTE IMPORTANTE

Faire des dessins avec la METHODE des instructions PRINT ne permet pas d'incorporer des caractères en CONTRASTE INVERSE (RVSon) pendant que tu fais ton dessin...c'est-à-dire lorsque tu n'as pas encore écrit tes numéros de ligne et ouvert les guillemets...

Mais tu peux les inclure après avoir entré ton dessin dans un programme. Si tu tiens compte de cette remarque, tu peux maintenant compléter ton dessin en ajoutant un balcon et un parterre:

Le balçon et le parterre

Avant de passer à la 4e étape, tu as une belle occasion d'insérer des caractères en contraste inversé... ce qui était impossible pendant la construction par la méthode des PRINT.

Ecris LIST et appuie RETURN. Enlève l'instruction 1190 GOTO 1190. Tu n'as qu'à écrire 1190 et taper RETURN. Puis tu transformes la dernière ligne de ton programme (no 1170) de la façon suivante:

1- Remonte le curseur sur la ligne 1170 et conduis-le vers la droite jusqu'à ce qu'il soit sur les guillemets. Retape les guillemets (SHIFT 2) afin que les caractères de commande d'affichage (page 35) puissent s'imprimer sur l'écran.

2- Appuie sur la barre d'espacement jusqu'à ce que le curseur soit sur le caractère situé au coin des fondations.

3- Appuie simultanément sur CTRL et RVSon, C= et V, espace cinq fois, simultanément sur CTRL et RVSoff, C= et T cinq fois, C= et U deux fois, SHIFT M puis RETURN.

Tu viens de dessiner la 1re marche et le balcon en caractères inversés. Si READY est là, tu l'effaces et tu écris:

1180 PRINT" ↑
└tu appuies la barre d'espacement pour amener le curseur sur la colonne qui précède le caractère RVSon imprimé dans l'instruction précédente.

Ensuite tu appuies simultanément sur CTRL et RVSon, C= B, simultanément sur CTRL et RVSoff, espace 13 fois, signe moins, SHIFT M, puis RETURN.

Tu viens de dessiner deux marches en caractères inversés et une partie du parterre.

1190 PRINT" ↑
└tu appuies...... comme ci-dessus en 1180...

Ensuite tu tapes 18 fois C= et U puis RETURN.

1200 GOTO 1200 (pour empêcher l'apparition de READY)

Ton dessin est terminé. Ecris RUN et RETURN. Si les marches sont en "bon état", tu peux réaliser la 4e étape de ton projet.

Quatrième et dernière étape

Tu dois maintenant sauvegarder ton programme sur cassette ou disquette car il faudra le compléter avec des instructions POKE...et un SPRITE dans une prochaine "activité..."

A) Avec une CASSETTE:

1- Ecris: SAVE" ↑ "
—inscris entre guillemets le nom que portera ton programme.

2- Mets le magnétocassette en marche en appuyant sur les 2 touches pour "enregistrement" (record and play)

3- Quand c'est terminé, tu rebobines le ruban magnétique pour avoir le début de ton programme.

4- Tu vérifies si ton programme est bien implanté sur la cassette:

Ecris: VERIFY" ↑ "
—inscris le nom de ton programme

5- Mets le magnéto en marche en appuyant sur une seule touche pour la lecture (play).

6- Si tu veux une preuve évidente que ton programme est bien sur la cassette:
a) Ecris: NEW et appuie sur RETURN (cette opération efface toutes les mémoires).
b) Retrace le début de ton programme déjà inscrit sur la bande magnétique.
c) Ecris LOAD et le nom de ton programme entre guillemets puis RETURN. La commande LOAD introduit ton programme dans la mémoire de ton ordinateur... elle "charge" ton programme.

Sur l'écran tu lis la phrase suivante: PRESS PLAY ON TAPE. Donc appuie sur la touche lecture du magnéto (play).

7- Enfin, écris: RUN et appuie sur RETURN pour admirer ton chef-d'oeuvre sur l'écran!

B) Avec une DISQUETTE:

1- Ecris: SAVE" ↑ ",8
└──────inscris le nom de ton programme

2- Ecris: LOAD" ↑ ",8
└──────inscris le nom de ton programme

Le chiffre 8 placé après la virgule est le No de l'unité de disque (ex. la 1541). Le No 1 est réservé pour ton magnétocassette...mais tu n'es pas obligé de l'écrire car le C64 s'en charge (écriture 'par défaut').

SOLUTION détaillée de la première étape (de la page 24).

Voici les différentes touches à appuyer pour obtenir le dessin de la maison. Voir la p.26 pour le balcon et le parterre.

Commence par effacer l'écran: SHIFT et CLR/HOME. Ensuite descends le curseur sur la rangée3 (n'oublie pas de compter la rangée 0,1,2,3). Déplace le curseur vers la droite, sur la colonne 13 (n'oublie pas de compter la colonne 0,1,2,3,...13). Ainsi ton dessin sera construit à droite de l'écran car il sera décalé vers la gauche au moment de l'exécution du programme.

Rangée 3: colonne 13, appuie sur SHIFT et M, espace, SHIFT N.
Rangée 4: colonne 12, SHIFT étoile, espace, SHIFT Q, espace, SHIFT étoile.
Rangée 5: colonne 13, appuie SHIFT et M, espace, SHIFT N.
Descends le curseur sur la rangée 8 (rien sur les rangées 6 et 7).
Rangée 8: colonne 21, appuie sur ↑ .
Rangée 9: colonne 21, appuie sur C= et signe plus, C= et T 5 fois, SHIFT M.
Rangée 10: colonne 14, SHIFT N, C= T 12 fois, SHIFT M 2 fois
Rangée 11: colonne 13, SHIFT N, 14 espaces, SHIFT M 2 fois, signe moins.
Rangée 12: colonne 12, SHIFT N, 16 espaces, SHIFT M 2 fois.
Rangée 13: colonne 11, SHIFT N, 18 espaces, SHIFT M, C= G, signe moins.
Rangée 14: colonne 11, C= T, SHIFT O, C= T 17 fois, SHIFT O, C= G.
Rangée 15: colonne 10, SHIFT U, SHIFT étoile, C= G, SHIFT U, SHIFT étoile 2 fois, SHIFT I, 4 espaces, C= D, C= I 4 fois, C= F, 3 espaces, C= G 2 fois.
Rangée 16: colonne 10, SHIFT Q, espace, C= G, SHIFT signe moins, C= A, C= S, SHIFT signe moins, 4 espaces, C= L,
→RVSon, 4 espaces, RVSoff, C= J, 3 espaces, C= G 2fois

Note: [mettre] RVSon...lorsque tu feras le balcon et le parterre
Rangée 17: colonne 12, C= G, SHIFT signe moins, C= Z, C= X, SHIFT signe moins, 4 espaces, C= L,
→RVSon, 4 espaces, RVSoff, C= J, 3 espaces, C= G 2 fois

Rangée 18: colonne 12, C= G, SHIFT -, C= V. espace, SHIFT -,
4 espaces, C= U 6 fois, 3 espaces, C= G 2fois.
Rangée 19: colonne 12, C= G, SHIFT -, 2 espaces, SHIFT -,
13 espaces, C= G 2 fois, signe -
Rangée 20: colonne 12, C= G, SHIFT -, 2 espaces,
SHIFT -, 13 espaces, C= G 2 fois.
Rangée 21: colonne 11, C= T 19 fois, C= U 2 fois.

<u>Des CONSEILS pratiques:</u>

1- Quand tu es en "<u>mode</u> quillemet", les touches CRSR affichent les caractères de commande 3,4,5,6 (droite,gauche,bas,haut, page 35) et la touche INST affiche le caractère 10 (p.35)

Etre en "mode guillemet" veut dire que tu écris **pendant que les quillemets sont ouverts.**

Si tu veux corriger une erreur en te servant des touches CRSR et INST, tu dois d'abord **sortir du mode guillemet** <u>soit en fermant</u> les guillemets (que tu effaces ensuite avec la touche DEL) <u>soit en revenant</u> dans la ligne après avoir tapé RETURN.

2- Si ton dessin apparaît trop à gauche quand tu fais RUN, tu peux le déplacer vers la droite:

Tu écris LIST et RETURN puis tu arrêtes le déroulement de l'écran avec la touche STOP.

Tu amènes ensuite le curseur <u>après les guillemets</u> de <u>chacun</u> des PRINT (du dessin) et tu appuies sur SHIFT et INST une ou deux ou... fois. N'oublie pas de taper RETURN pour que ta correction se place en mémoire.

3- Quand ton dessin est sur l'écran, tu <u>peux voir</u> les <u>lettres</u> des caractères graphiques que tu as utilisés dans ton dessin <u>si tu appuies</u> simultanément sur: C= <u>et</u> SHIFT.

4- Plus tard tu pourras embellir ta maison, la peinturer, changer de toiture...Tu pourras ajouter quelques nuages, planter un arbre, mettre des fleurs...

Je te conseille d'attendre avant de "trop investir" car les chapitres qui suivent t'enseigneront de nouvelles techniques, de nouveaux matériaux pour économiser "temps et argent". Pour le moment <u>tu as un toit!</u>...

Alors commence la construction de ta fusée...

PROJET B-1

Dessine une fusée. Fais d'abord ton dessin sur une feuille quadrillée (voir le modèle page 72).

Entre ton dessin dans un programme par la <u>méthode des PRINT</u>

Implante ton programme sur bande magnétique ou sur disquette.

SOLUTION: pages 67 et 72.

PROJET C-1

Dessine 7 oiseaux regroupés en "V" avec la pointe du V orientée vers la droite de l'écran.

Entre ton dessin dans un programme par la <u>méthode des PRINT</u>

Sauvegarde ton programme sur cassette ou sur disquette.

SOLUTION: page 33.

<u>QUESTION 5</u> Quoi faire pour que l'ensemble des oiseaux soit 2 lignes <u>plus bas</u> sur l'écran?

Réponse: Tu ajoutes une instruction 95 dans le programme C-1 de la page 33. Exemple: 95 PRINT:PRINT
Ou bien tu ajoutes 2 caractères de commande <u>No 5</u> p.35 dans les guillemets de la ligne 90.

PROJET D-1

Dessine une **<u>silhouette</u>** du Québec.

Pour ce projet D-1, il faut procéder différemment car une <u>silhouette</u> s'obtient avec des <u>caractères en contraste inversé</u> (comme pour le balcon et l'escalier de la maison).

<u>Je te conseille</u>
1- d'utiliser une feuille quadrillée (grandeur nature du fond de l'écran d'une TV 14"), et d'y superposer une carte du Québec (voir le modèle page 34).

2- tu cherches ensuite, ligne par ligne, les caractères qui ressemblent le plus aux contours à reproduire. Pour t'aider dans ce choix des caractères, regarde le <u>listing</u> du programme D-1 page 33 <u>ou</u> la <u>version offerte à la p. 34</u>.

<u>Remarque</u> sur les <u>contours</u>... Les contours offriront ici une précision assez médiocre car nous sommes en "faible résolution" (une résolution de 40 X 25 i.e. 1000). Nous pourrons faire

beaucoup mieux dans le prochain volume avec la "haute résolution" (40 X 8) X (25 X 8) i.e. 64000.

1- **Une instruction** (anglais: Statement) est un ordre que le programmeur donne à l'ordinateur.

Le C64 possède 23 instructions pour le langage BASIC (avec les périphériques TV et CASSETTE).

Tu connais déjà deux instructions: GOTO et PRINT (en abrégé: ?)

2- **Une commande** (anglais: Command) est un énoncé qui permet une conversation directe avec l'ordinateur pour lui dire quoi faire avec un programme.

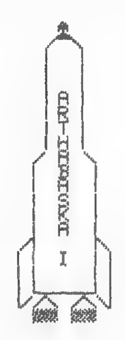
Tu en as 7 à apprendre CONT, LIST, LOAD, NEW, RUN SAVE et VERIFY. Tu les connais déjà, sauf "CONT" que tu apprendras lors de la prochaine "activité dirigée".

3- **Une fonction** bibliothèque est un petit programme utilitaire que le C64 garde en mémoire et auquel tu accèdes simplement en signalant le nom de la fonction. Le C64 possède 28 fonctions...Voir chapitre IV...

SOLUTION du PROJET B-1 de la page 31.

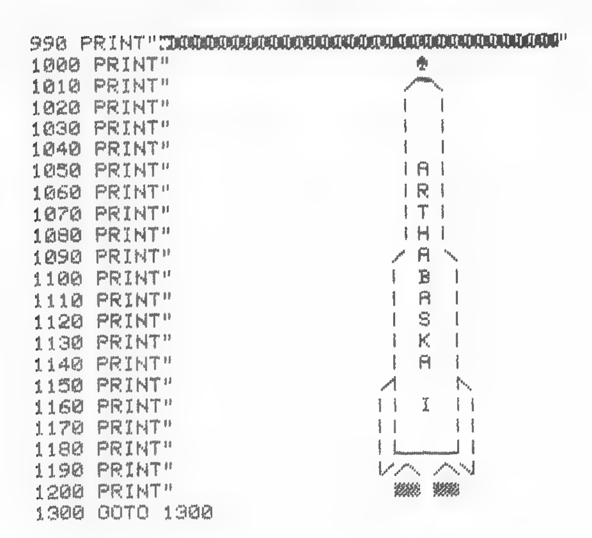
Programme B-1

avec RUN et RETURN



Listing du programme B-1

avec LIST et RETURN



```
990 PRINT"[illegible]"
1000 PRINT"                    ♠
1010 PRINT"                   /‾\
1020 PRINT"                   | |
1030 PRINT"                   | |
1040 PRINT"                   | |
1050 PRINT"                   |A|
1060 PRINT"                   |R|
1070 PRINT"                   |T|
1080 PRINT"                   |H|
1090 PRINT"                  /A \
1100 PRINT"                  | B |
1110 PRINT"                  | A |
1120 PRINT"                  | S |
1130 PRINT"                  | K |
1140 PRINT"                  | A |
1150 PRINT"                 /|   |\
1160 PRINT"                 || I ||
1170 PRINT"                 ||   ||
1180 PRINT"                 ||___||
1190 PRINT"                 \/\ /\/
1200 PRINT"                  ▒▒ ▒▒
1300 GOTO 1300
```

SOLUTIONS des PROJETS C-1 et D-1 de la page 31

Programme C-1

Listing du programme C-1

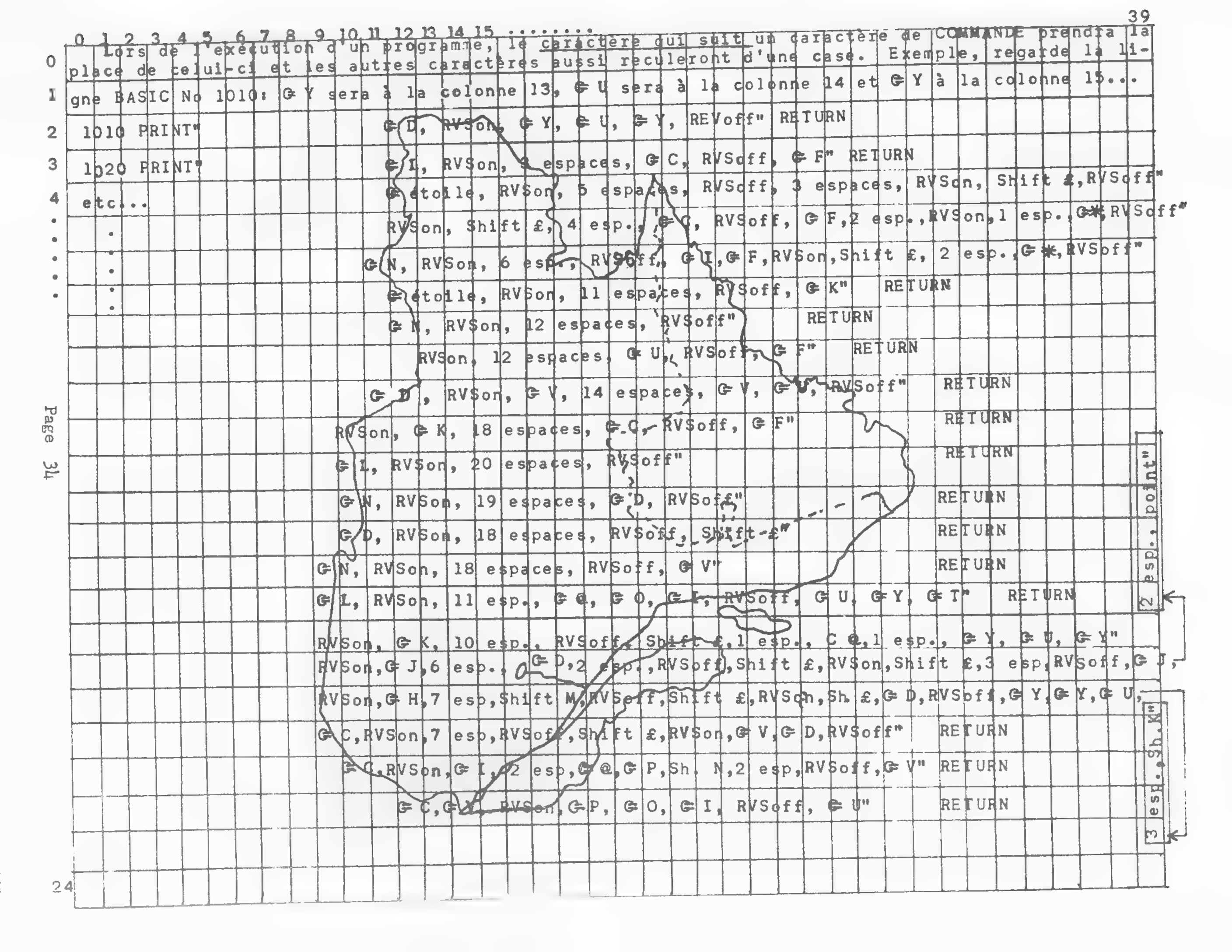
```
95 PRINT"□"
100 PRINT"
110 PRINT"
120 PRINT"
130 PRINT"
140 PRINT"
150 PRINT"
160 PRINT"
170 PRINT"
180 PRINT"
190 PRINT"
200 PRINT"
210 PRINT"
220 PRINT"
230 PRINT"
240 PRINT"
250 GOTO 250
```

Programme D-1

Listing du programme D-1

```
990 PRINT"□"
1000 PRINT
1010 PRINT"
1020 PRINT"
1030 PRINT"
1040 PRINT"
1050 PRINT"
1060 PRINT"
1070 PRINT"
1080 PRINT"
1090 PRINT"
1100 PRINT"
1110 PRINT"
1120 PRINT"
1130 PRINT"
1140 PRINT"
1150 PRINT"
1160 PRINT"
1170 PRINT"
1180 PRINT"
1190 PRINT"
1200 PRINT"
1210 PRINT"
1230 GOTO 1230
```

Plus tard nous apporterons plus de respect aux contours de notre Québec en travaillant en haute résolution.



Lors de l'exécution d'un programme, le caractère qui suit un caractère de COMMANDE prendra la place de celui-ci et les autres caractères aussi reculeront d'une case. Exemple, regarde la ligne BASIC No 1010: C= Y sera à la colonne 13, C= U sera à la colonne 14 et C= Y à la colonne 15...

```
1010 PRINT"          C= D, RVSon, C= Y, C= U, C= Y, REVoff" RETURN
1020 PRINT"          C= I, RVSon, 3 espaces, C= C, RVSoff, C= F" RETURN
etc...               C= étoile, RVSon, 5 espaces, RVSoff, 3 espaces, RVSon, Shift £,RVSoff"
  .                  RVSon, Shift £, 4 esp., C= C, RVSoff, C= F,2 esp.,RVSon,1 esp.,C= *,RVSoff"
  .                C= N, RVSon, 6 esp., RVSoff, C= I,C= F,RVSon,Shift £, 2 esp.,C= *,RVSoff"
  .                  C= étoile, RVSon, 11 espaces, RVSoff, C= K"   RETURN
  .                  C= N, RVSon, 12 espaces, RVSoff"   RETURN
  .                     RVSon, 12 espaces, C= U, RVSoff, C= F"   RETURN
                   C= D, RVSon, C= V, 14 espaces, C= V, C= U, RVSoff"   RETURN
                 RVSon, C= K, 18 espaces, C= C, RVSoff, C= F"   RETURN
                 C= I, RVSon, 20 espaces, RVSoff"   RETURN
                 C= N, RVSon, 19 espaces, C= D, RVSoff"   RETURN
                 C= D, RVSon, 18 espaces, RVSoff, Shift £"   RETURN
               C= N, RVSon, 18 espaces, RVSoff, C= V"   RETURN
               C= L, RVSon, 11 esp., C= @, C= O, C= I, RVSoff, C= U, C= Y, C= T"   RETURN
               RVSon, C= K, 10 esp., RVSoff, Shift £,1 esp., C @,1 esp., C= Y, C= U, C= Y"
               RVSon,C= J,6 esp., C= D,2 esp.,RVSoff,Shift £,RVSon,Shift £,3 esp,RVSoff,C= J,
               RVSon,C= H,7 esp,Shift M,RVSoff,Shift £,RVSon,Sh. £,C= D,RVSoff,C= Y,C= Y,C= U,
               C= C,RVSon,7 esp,RVSoff,Shift £,RVSon,C= V,C= D,RVSoff"   RETURN
                 C= C,RVSon,C= I,2 esp,C= @,C= P,Sh. N,2 esp,RVSoff,C= V"   RETURN
                    C= C,C= Y, RVSon,C= P, C= O, C= I, RVSoff, C= U"   RETURN
```

2 esp. . point"

3 esp. Sh. K"

Voici le symbole et le rôle que joue chacun des onze

CARACTERES de COMMANDE d'AFFICHAGE du C64

Ces caractères spéciaux apparaissent lorsque tu ouvres les guillemets et que tu tapes les touches suivantes: (en mode direct ou programmé):

NOS	Les touches	Code ASCII CHR$ ( )	Représentation du caractère d'affichage	Son rôle
1-	SHIFT et CLR HOME	147		Vider (effacer l'écran)
2-	CLR HOME sans SHIFT	19	S	Curseur à l'origine (en haut à gauche) Sans vider l'écran.
3-	CRSR	29		Curseur à droite.
4-	SHIFT et CRSR	157		Curseur à gauche.
5-	CRSR	17	Q	Curseur en bas.
6-	SHIFT et CRSR	145		Curseur en haut.
7-	CTRL et RVS ON	18	R	Inversion du contraste (caractère bleu foncé sur fond bleu **clair**)
8-	CTRL et RVS OFF	146		Retour au contraste normal (caractère bleu **clair** sur fond **bleu foncé**
9-	INST DEL sans SHIFT	20	T	Supprimer un caractère
10-	SHIFT et INST DEL	148		Insérer un caractère
11-	BARRE D'ESPACEMENT	32 ou 160		**Faire un espace. Si tu es en RVSon, l'espace sera bleu clair.**

CHAPITRE II

ACTIVITES DIRIGEES Série no 2

BUTS

Cette deuxième série d'activités te permettra d'apprendre:

1- Cinq nouvelles instructions: FOR... TO... STEP, NEXT, GET, IF... THEN, POKE.

2- La commande: CONT (tu connaîtras ainsi l'ensemble des 7 commandes du C64.

3- Cinq nouveaux caractères de "COMMANDE d'AFFICHAGE": voir les numéros 2, 3, 4, 5, 6, page ? .

4- Le tableau des codes-caractères d'écran (page 75)

5- Le tableau des codes-couleurs d'écran (page 38)

6- Les variables numériques et alphanumériques (ou chaînes de caractères)

7- Les six opérateurs relationnels (ou comparatifs)

8- Les ordinogrammes (ou organigrammes)

9- Le calcul binaire

10- Les SPRITES: Comment les introduire dans la mémoire du C64 et les déplacer sur l'écran

11- Comment utiliser l'horloge du C64.

POKE

Tu peux afficher des caractères sur l'écran par des PRINT ou par des POKE.

De même, tu peux mettre des couleurs sur l'écran par des PRINT (niveau élémentaire) ou par des POKE (niveau élaboré).

Mais POKE nous offre l'avantage d'écrire directement une valeur dans une mémoire vive: MEV.

Il y a deux blocs mémoires d'écran et dans chacune d'elles tu peux placer une valeur codée avec POKE.

Il y a un bloc de 1000 mémoires pour les caractères (1000 octets). Voir les codes page 75.

Il y a un bloc de 1000 mémoires pour les couleurs (1000 quartets). Voir les codes couleur page 38.

Les 1000 mémoires d'écran des CODES-CARACTERES
(ou matrice vidéo)

La mémoire des caractères contient le code (0 à 255) de chaque caractère à afficher. Le code binaire est sur 8 bits: 2↑8 = 256 en décimal (2↑8 correspond à 2 puissance 8).

Ce bloc mémoires commence à l'adresse 1024 et se termine à l'adresse 2023.

EXERCICE 1 Appuie à la fois sur les touches RUN/STOP et RESTORE. Ecris en mode direct:

```
POKE 1024,83
```

et tape RETURN

Tu viens de placer le caractère coeur (voir code 83, page 76) à l'adresse 1024, la case du coin en haut à gauche

(origine). Le caractère est invisible parce que tu n'as pas mis de couleur; il est bleu foncé sur fond bleu foncé

Monte le curseur à l'origine et tu apercevras le caractère coeur pendant que le curseur clignote.

Tu peux rendre visibles les caractères placés sur l'écran avec POKE sans utiliser les mémoires d'écran des CODES-COULEURS Il suffit de changer la couleur bleu foncé du fond de l'écran.

EXERCICE 2 Ecris en mode programmé

```
10 POKE 1524,83
20 POKE 2023,83
30 POKE 53281,1
```

écris RUN et tape RETURN

L'instruction 30 change la couleur du fond de l'écran et te permet d'apercevoir les 3 caractères coeurs... Ce registre (mémoire) 53281 sera étudié plus tard.

EXERCICE 3 Comment apporter un changement dans ton programme:

Remonte le curseur sur la ligne 10 et déplace-le vers la droite jusqu'à ce qu'il coincide avec le chiffre 3. Tape sur le chiffre 1 et appuie RETURN. Fais la même chose sur le chiffre 3 de la ligne 20. Ensuite, fais exécuter ton programme que tu viens de transformer (RUN et RETURN).

C'est maintenant une balle qui apparaît au centre et au bas de l'écran (voir code 81, page 75).

Il est important de noter que c'est ce code 81 (en binaire) qui s'inscrit dans la case mémoire... Les adresses 1524 et 2023 sont attribuées à des cases un peu comme les numéros inscrits au-dessus des portes des maisons.

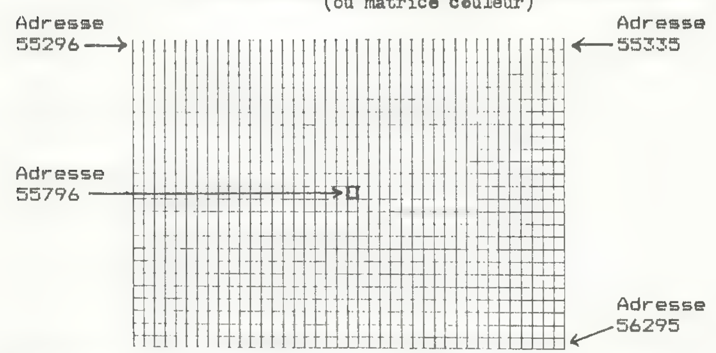
Tableau des CODES-COULEURS d'écran

Avec POKE	COULEUR	Avec POKE	COULEUR
0	Noir	8	Orange
1	Blanc	9	Brun
2	Rouge	10	Rouge clair
3	Turquoise	11	Gris foncé
4	Pourpre	12	Gris perle
5	Vert	13	Vert clair
6	Bleu	14	Bleu clair
7	Jaune	15	Gris clair

La mémoire d'écran des couleurs contient le code (0 à 15) de chaque couleur à afficher. Le code binaire est donc sur 4 bits. Si tous les bits sont à 1 tu auras le code 15 (gris clair).

Le bloc mémoires d'écran commence à l'adresse 55296 et se termine à l'adresse 56295.

Les **1000 mémoires d'écran des CODE-COULEURS**
(ou matrice couleur)



EXERCICE 4 Mets un coeur rouge au centre de l'écran. (En mode direct)

```
POKE 1524,83: POKE 55796,2        tape RETURN
```

On dit: Le caractère coeur est à l'adresse 1524 et son code écran est 83 (voir tableau page 75). La couleur rouge est à l'adresse 55796 et son code-couleur est 2 (tabl.p.38).

EXERCICE 5-A Utilise l'instruction POKE pour dessiner un soleil orange à l'adresse 1158 et 6 rayons jaunes

```
10 PRINT"□"
20 POKE 1117,77: POKE 55389,7: POKE 1119,78: POKE 55391,7
30 POKE 1156,67: POKE 55428,7: POKE 1158,81: POKE 55430,8
40 POKE 1160,67: POKE 55432,7: POKE 1197,78: POKE 55469,7
50 POKE 1199,77: POKE 55471,7
```

EXERCICE 5-B Refais l'exercice 5-A et utilise la variable E = 1024 pour la mémoire d'écran située à l'origine (0,0).
Et la variable C = 55296 pour la mémoire d'écran située à l'origine (0,0).

```
10 E=1024: C=55296
20 POKE E+93,77:POKE C+93,7:POKE E+95,78:POKE C+95,7
30 POKE E+132,67:POKE C+132,7:POKE E+134,81:POKE C+134,8
40 POKE E+136,67:POKE C+136,7:POKE E+173,78:POKE C+173,7
50 POKE E+175,77:POKE C+175,7
```

Le code de la couleur jaune est 7. Le tableau, page 38, te donne les codes des 16 couleurs du C64 que tu peux obtenir avec POKE. Le tableau des CODES-CARACTERES d'ECRAN est à la page 75.

A l'aide de ces 2 tableaux, <u>invente plusieurs exercices</u> pour te familiariser avec l'instruction POKE.

Tu vas étudier maintenant trois instructions fondamentales: FOR... TO, NEXT et IF... THEN

FOR... TO... NEXT

L'avantage d'une machine c'est de pouvoir lui faire exécuter des travaux répétitifs. C'est ce que fait l'ordinateur dans une boucle FOR... NEXT.

EXERCICE 6-A Fais imprimer 16 fois le caractère <u>coeur</u> sur une ligne verticale

```
5 PRINT"□"
10 FOR N=1 TO 16
20 PRINT"♥"
30 NEXT N
```

Ecris RUN et appuie RETURN et tu seras "cordialement" comblé

Que se passe-t-il au juste? La lettre N est une <u>variable numérique</u> qui va prendre successivement les valeurs 1, 2, 3... jusqu'à 16.

A chaque <u>itération</u>, c'est-à-dire chaque fois que l'ordinateur rencontre l'instruction NEXT N, il ajoute 1 à la variable N puis il recommence la boucle. Ca paraît plus "savant" quand tu dis: La variable de contrôle N est incrémentée de 1 à chaque itération.

<u>Il faut noter</u> que la variable N <u>forme l'adresse</u> d'un emplacement mémoire. Tu peux prendre <u>une autre lettre</u> que N comme variable. En fait le C64 te permet d'utiliser les identificateurs de variables suivants:

Soit une lettre: Exemple A, B, C,... Z.
Soit une lettre et un chiffre (0à9): Exemple A0, A1, B2, N1...
Soit deux lettres: Exemple AA, BB, CC, NN, AC, BD,...
(exceptions: ST et TI)

Essaie... Remplace N par NA ou par N2...

<u>Quand N vaut 16</u> l'ordinateur <u>sort de la boucle</u> et exécute l'instruction qui vient après NEXT. S'il n'y a pas d'instruction il considère le programme comme terminé (END) et il écrit READY.

NOTE: Lorsqu'il n'y a pas d'ambiguité, tu peux écrire **seulement NEXT** au lieu de NEXT suivi de la variable. Nous ferons souvent ainsi dans les prochains exercices.

Il serait bon que tu connaisses dès maintenant le rôle du mot-clef <u>STEP</u> (le pas).

EXERCICE 6-B En principe, la ligne 10 de l'exercice 6-A aurait dû être écrite ainsi:

```
10 FOR N=1 TO 16 STEP 1
```

mais quand le pas (STEP) est égal à 1, le mot-clef STEP n'est pas obligatoire.

Si tu fais des demi-pas, tu écris STEP .5 (note: les ordinateurs fonctionnent encore avec le point au lieu de la virgule pour indiquer les décimales).

EXERCICE 6-C Ecris la ligne 10 de l'exercice 6-A comme ceci:

```
10 FOR N=.5 TO 8 STEP .5
```

et tu obtiendras encore 16 coeurs car l'ordinateur fera le même nombre d'itérations: .5, 1, 1.5, 2, 2.5, ... 8.

A la première itération, la valeur N doit être .5 sinon il y aura seulement 15 itérations (15 coeurs).

Tu peux mettre des espaces dans ta ligne BASIC afin de faciliter la lecture du programme, mais chaque espace occupe une case mémoire!

Enfin, quand tu commences par la plus grande valeur de la variable N, tu dois mettre un signe moins devant le pas. Ex.: -.5

EXERCICE 7 Utilise un PAS de -.5 pour imprimer 16 coeurs.

```
10 FOR N=8 TO .5 STEP -.5
20 PRINT "♥"
30 NEXT
```

Tu auras encore 16 coeurs sur l'écran.
(SHIFT S)

Avant de passer à l'étude de IF...THEN, ajoute un ; à la fin de la ligne 20:

```
10 FOR N=1 TO 16
20 PRINT "♥";
30 NEXT
```

Les 16 coeurs s'impriment les uns à la suite des autres. Donc le caractère ; empêche le curseur de retourner à la ligne. On dit: il n'y a pas de "retour chariot" (partie d'une machine à écrire qui porte le rouleau).

IF...THEN

L'instruction IF... THEN permet de tester une condition, c'est-à-dire de déterminer SI elle est VRAIE ou FAUSSE.

Si elle est vrai ALORS — aller à une certaine ligne du programme ou écrire l'instruction (ou les instructions) après THEN.

EXERCICE 8 Dans ton programme qui imprime 16 coeurs verticalement sur 16 lignes, ajoute une instruction IF...THEN pour sortir de la boucle après avoir imprimé 8 coeurs.

```
10 FOR X=1 TO 16
20 PRINT " ♥"
25 IF X=8 THEN END
30 NEXT
```

← après avoir imprimé 8 coeurs, l'ordinateur imprimera READY, i.e. qu'il est prêt à faire autre chose..

Attention: ici le IF doit être après la ligne 20. Si tu mets:

```
15 IF X = 8 THEN END
```

← tu auras seulement 7 coeurs sur l'écran. Pourquoi?...

Si pour une raison quelconque tu veux mettre le **IF** à la ligne 15 et avoir quand même 8 coeurs, il faudra écrire:

```
15 IF X > 8 THEN END
```

(Le symbole > veut dire: supérieur à)

Les opérateurs relationnels (ou comparatifs)

- = égal
- < inférieur
- > supérieur
- >= supérieur ou égal
- <= inférieur ou égal
- <> différent

En BASIC il y en a 6

NOTE: Ces opérateurs peuvent aussi être utilisés avec des variables chaînes.

EXERCICE 9 Tu veux imprimer un coeur au début de chaque rangée de l'écran, donc 25 coeurs. Ecris le programme suivant:

```
10 FOR X=1 TO 25
20 PRINT"♥ "
25 NEXT
30 GOTO 30
```

Tu constates qu'il n'y a que 24 coeurs! L'impression du 25e coeur a fait dérouler l'écran et le premier a disparu. Quoi faire pour éviter ces "haut-le-coeur"?

EXERCICE 10 Une première solution pour empêcher un retour chariot: mets un ; après le 25 e PRINT.

```
10 FOR X=1 TO 25
```

```
15 IF X= 25 THEN PRINT"♥";:GOTO 30
20 PRINT"♥"
25 NEXT X
30 GOTO 30
```

Tu vois que SI X= 25 la ligne 15 fait imprimer le 25e coeur sans "retour chariot" (à cause du ;). De plus, la ligne 15 fait sortir de la boucle par l'instruction GOTO afin d'empêcher que la ligne 20 vienne imprimer un autre coeur à droite du 25e et provoquer un retour chariot.

EXERCICE 11 Une deuxième solution pour empêcher un retour chariot: remonter le curseur d'une ligne avant que l'opération "retour chariot" ait lieu.

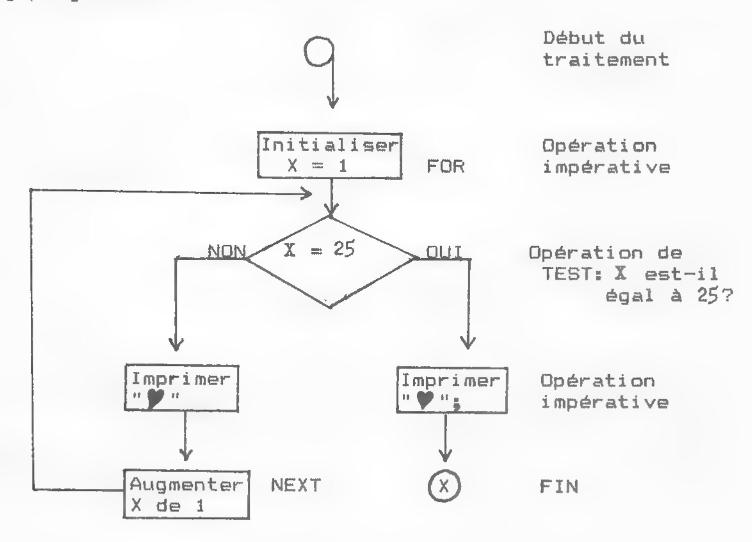
```
10 FOR X=1 TO 25
15 IF X=25 THEN PRINT"♥]":GOTO 30
20 PRINT"♥"
25 NEXT X
30 GOTO 30
```

← Ce caractère de COMMANDE d'AFFICHAGE s'obtient avec SHIFT et CRSR (voir no 6 page 35).

Il y a une 3e solution avec la fonction PEEK que tu étudieras plus loin.

Voici ce que donnerait **l'ordinogramme (encore appelé organigramme)** du petit programme ci-dessus: (exercice 10)

Je te signale qu'un organigramme est une représentation graphique qui permet de documenter une partie d'un programme ou de l'expliquer. Il est de moins en moins utilisé pour traduire un long programme:



VARIABLES CHAINES de CARACTERES

Tu sais déjà qu'une variable numérique représente un nombre (p.40). Exemple:
10 E=1024: C=55296

La variable numérique E représente le nombre 1024 et C représente 55296. Cette ligne BASIC no 10 se lit comme suit: mets la valeur 1024 dans la variable E et la valeur 55296 dans la variable C.

NOTE Le caractère [:] indique toujours le début d'une nouvelle instruction.

Il y a aussi des variables appelées **variables chaînes de caractères** (anglais: string). Le nom de ces variables s'imprime comme celui des variables numériques et se termine toujours par le signe [$].

Identification des variables chaînes de caractères:

Pour identifier une variable chaîne, tu peux utiliser soit une lettre: A$, B$, C$,... Z$.
soit une lettre et un chiffre: A0$, A1$, B2$, G3$,...
soit deux lettres: AA$, AB$, AZ$, BD$, HI$...(exception: TI$)

Ces variables (avec $) peuvent représenter n'importe quel caractère que le C64 peut imprimer sur l'écran: lettres, chiffres, graphiques, contrôles curseur... Lorsque ces caractères sont **placés entre guillemets**, ils forment une **chaîne de caractères** (ou simplement: une chaîne).

Voici 4 exemples de chaînes de caractères: "TEL:357-2082" "PAUL DUPONT" "ARTHABASKA G6P 5W1" "905 BOUL. BOIS-FRANCS"

EXERCICE 12 Mets la valeur de chacune de ces 4 chaînes dans une variable et fais imprimer le contenu de ces quatre variables dans l'ordre d'une adresse.

```
10 A$="TEL:357-2082"
20 B$="JOSEPH DUPONT"
30 C$="ARTHABASKA G6P 5W1"
40 D$="905 BOUL. BOIS-FRANCS"
50 PRINT B$:PRINT D$:PRINT C$:PRINT A$              RUN et RETURN
```

Une chaîne peut contenir jusqu'à 255 caractères...mais rappelle-toi que tu ne peux mettre plus de 80 caractères par ligne BASIC (y compris les caractères espaces)...**Et une ligne BASIC occupe 2 lignes sur l'écran du C64**. Exemple:

EXERCICE 13 Voici 1 ligne de 80 caractères (**2 lignes sur écran**)

```
10 A$="OH CHERE MARQUISE TES GRANDS YEUX NOIRS NOUS FONT MOURIR
   D'AMOUR":PRINTA$
```

Une chaîne de caractères pourrait contenir seulement <u>un</u> caractère et on l'appellerait encore chaîne. Exemple B$="N" ou encore A$=" " où la chaîne contient le caractère espace.

Une chaîne peut aussi ne contenir <u>aucun</u> caractère c'est-à-dire que les guillemets se suivent. Exemple C$="". On l'appelle <u>chaîne vide</u>. C'est une chaîne <u>vide</u> ou une chaîne <u>de un</u> caractère qu'on utilise <u>avec GET</u> (voir ci-dessous).

GET

L'instruction GET (obtenir) est placée dans un programme pour permettre à l'ordinateur de <u>"recueillir" le caractère</u> que l'utilisateur tapera sur le clavier <u>et de mettre</u> ce caractère dans la variable qui suit GET (GET lit <u>un</u> caractère à la fois).

Il faut donc que l'ordinateur <u>attende</u> ce caractère, c'est pourquoi GET est presque toujours accompagné de l'instruction IF... THEN.

<u>EXERCICE 14</u> Ecris un petit programme qui provoque l'ATTENTE d'une touche quelconque du clavier.

```
10 PRINT"APPUIE UNE TOUCHE"
20 GET C$:IF C$="" THEN 20
30 PRINT C$       └──────────(la chaîne C$ est vide)
40 END
```

Si aucun caractère n'est tapé, la chaîne reste vide et l'ordinateur "tourne en rond" dans la ligne 20 comme dans une boucle infini. Ici la boucle s'appelle **<u>boucle de scrutation</u>** car l'ordinateur scrute (teste) l'ensemble du clavier pour vérifier si une touche a été enfoncée.

Si tu tapes un caractère quelconque, l'ordinateur met ce caractère dans la <u>variable</u> qui suit GET (dans le jargon de l'informatique on dit que GET extrait le premier caractère du <u>tampon clavier</u>...là où la valeur <u>ASCII</u> du caractère tapé au clavier va se loger).

ASCII: <u>A</u>merican <u>S</u>tandard for <u>C</u>ommunication <u>I</u>nformation <u>I</u>nterchange (voir p. 76).

Puis l'exécution du programme se continue en imprimant le <u>contenu</u> de la variable C$, c'est-à-dire le caractère que tu as appuyé.

Avec une boucle de délai (d'attente), **<u>le temps d'attente est fixé par le programme</u>**. Exemple: FOR T=1 TO 1000:NEXT T provoquera une attente d'environ <u>1 seconde</u>.

Mais avec GET accompagné de IF **<u>c'est toi qui décides du temps</u>** de réflexion dont tu as besoin avant de répondre à telle ou telle question.

EXERCICE 15 Ecris un petit programme de choix de réponses
Est-ce que 9 X 6 est égal à 54?

```
10 PRINT"9 X 6 = 54 VRAI? OU FAUX?"
20 PRINT"APPUIE V OU F"
30 GET A$:IF A$="V" THEN 80
40 IF A$="F" THEN 60
50 GOTO 30
60 PRINT:PRINT A$" JE REGRETTE"
70 GOTO 90
80 PRINT:PRINT A$" FELICITATION!"
90 END
```

Aux lignes 30 et 40 l'ordinateur scrute le clavier pour savoir si tu as appuyé le caractère V (vrai) ou le caractère F (faux). Si aucun de ces 2 caractères n'a été tapé, la ligne 50 renvoie à la ligne 30 pour former une boucle de scrutation. Le seul moyen de sortir de la boucle (sans appuyer STOP), c'est de taper V ou F et tu obtiens la réponse soit à la ligne 80 ou à la ligne 60.

NOTES 1- Tu peux effacer l'écran et aérer l'affichage si tu utilisesles caractères de COMMANDE D'AFFICHAGE (page 35)
2- Avantages de GET par rapport à INPUT:
a) GET n'abîme pas le dessin sur l'écran car le curseur est invisible et il n'y a pas de point d'interrog.
b) Avec GET, pas besoin d'appuyer RETURN.

Les 4 caractères de commande d'affichage du CURSEUR

A la page 19 tu as fait des exercices avec les deux touches CRSR du curseur. En appuyant sur CRSR avec ou sans SHIFT tu déplaçais le curseur dans les 4 directions: droite--gauche et en bas--en haut.

Tu peux obtenir les mêmes résultats avec les 4 caractères de commande d'affichage du curseur: nos 3, 4, 5, 6 (voir tableau page 35).

Il faut que tu parviennes à reconnaître facilement ces symboles dans la liste d'un programme et évidemment, le rôle que joue chacun d'eux.

EXERCICE 16 Efface l'écran (SHIFT et CLR/HOME) et écris en mode direct

```
PRINT"QQQQQQQQQQQ]]]]]]]]]]]]]]]]]]JEAN"
```

JEAN apparaît au milieu de l'écran. Puisqu'il y a 11 Q en contraste inversé entre les guillemets, le mot JEAN s'imprimera 11 rangées plus bas que l'endroit où il aurait été imprimé si ces 11 Q inversés n'avaient pas été placés dans les guillemets.

De plus, il y a 18 crochets en contraste inversé entre les

guillemets, donc le mot JEAN se trouve déplacé de 18 colonnes (0-17) vers la droite.

EXERCICE 17 Efface l'écran et descends le curseur sur la rangée 10. Ecris en mode direct:

PRINT"▮▮▮[IIIIIIII]▲⬤◣" ←—(oiseau: SHIFT U, SHIFT Q, SHIFT I)

Laisse le curseur là où il est présentement, c'est-à-dire 3 rangées plus bas que l'oiseau. Tu écriras l'instruction de l'exercice 18 sur cette rangée.

Il y a 8 balles (SHIFT Q) en contraste inversé dans les guillemets, donc l'oiseau a été imprimé 8 rangées plus haut que l'endroit où il aurait été imprimé si ces balles inversées n'avaient pas été placées dans les guillemets. Tu sais déjà que les crochets inversés ont déplacé l'oiseau de trois colonnes vers la droite.

EXERCICE 18 Ecris, en mode direct, une instruction qui effacera l'oiseau.

PRINT"▮▮▮[III] ↑ " 3 espaces (avec la barre d'espaces)

L'oiseau disparaît de l'écran parce que 3 caractères éteints (espaces) ont été imprimés à l'endroit où l'oiseau se trouvait.

Comme par hasard, le curseur retombe au début de ton instruction! Descends-le sur l'instruction de l'exercice 18 et appuie RETURN. L'oiseau réapparaît à la même place ainsi que le curseur. Donc si tu appuies encore RETURN, l'oiseau disparaîtra de nouveau...

Ce déplacement manuel du curseur peut se faire automatiquement dans un programme (voir exercices 19 **et** 20).

EXERCICE 19 Ecris en mode programmé:

```
10 PRINT"*▮";
20 FOR I=1 TO 300:NEXT I
30 PRINT" ▮";
40 FOR I=1 TO 300:NEXT I
50 GOTO 10
```

Dans les guillemets des lignes 10 et 30, il y a un rectangle en contraste inversé avec une ligne blanche (éteinte) au centre. Le rôle de ce caractère de commande d'affichage est de faire reculer le curseur d'une colonne (case précédente) (voir no 4 page 35).

Il faut regarder chaque instruction PRINT d'un programme et pouvoir suivre le curseur "à la trace" et savoir exactement

sa position! Suivons-le ensemble comme un beau gibier qu'on ne voudrait pas perdre de vue:

L'instruction PRINT de la ligne BASIC no 10 dit 3 choses au C64:

1- fait imprimer une étoile au début de la rangée qui se trouve sous le mot RUN (le curseur se déplace sur la case suivante
2- recule le curseur sur la case précédente (c'est le rôle joué par le caractère no 4 page 35). Evidemment, ce "curseur fictif" n'efface pas l'étoile car tu sais que les "contrôles curseur" (CRSR) déplace le curseur sans détruire les caractères sur lesquels il se déplace...
3- imprime le premier caractère signalé dans la prochaine instruction PRINT à l'endroit où se trouve le curseur présentement (c'est le rôle du caractère ;).

Allons voir ce que nous dit la prochaine instruction PRINT. Elle se trouve à la ligne BASIC no 30 et elle demande aussi 3 choses au C64:

1- imprime le caractère espace à l'endroit où est positionné le curseur.
   L'étoile disparaît car le rôle du caractère espace est d'imprimer "un blanc" (effacer) en mettant tous les bits de la case où il se trouve à 0 (éteints). (Le curseur se déplace sur la case suivante).
2- et 3- comme ci-dessus...

Et le cycle recommence indéfiniment avec l'instruction GOTO 10. La ligne 20 forme une boucle d'attente. Le C64 revient 300 fois sur la même instruction, ce qui provoque une attente d'environ 1/3 de seconde. C'est suffisant pour que l'image impressionne la rétine de l'oeil! De même à la ligne 40, l'étoile disparaît (est effacée) pendant 1/3 de seconde.

EXERCICE 20 Refais l'exercice 19 en remplaçant l'étoile par un oiseau

```
10 PRINT"▟▖▌▌▌▌";
20 FOR I=1 TO 300: NEXT I
30 PRINT"   ▌▌▌▌";
40 FOR I=1 TO 300: NEXT I
50 GOTO 10
```

Le caractère de commande d'affichage "ORIGINE"

Il nous reste un dernier caractère de commande d'affichage à étudier, et ce n'est pas le dernier venu car il nous ramène à "notre origine"!

EXERCICE 21 Si le curseur est en haut de l'écran, descends-le vers le milieu et écris en mode direct:

```
PRINT"▒S"
```

Le chiffre 5 s'imprime en haut sur le coin gauche de l'écran (à l'origine). Le caractère de commande d'affichage <u>à l'origine</u> est symbolisé par un [S] en contraste inversé.

Pour obtenir ce caractère, il faut ouvrir les guillemets et appuyer sur HOME (sans SHIFT). Son rôle est de ramener <u>le curseur à l'origine</u>. (No 2 page 35)

<u>**EXERCICE 22**</u> Ecris en mode programmé:

```
5 PRINT"■"
10 PRINT"■■■■■■■■■■■■■■LA VIE"
20 PRINT"■■■■■■■■■■■■■■■■■■OU"
30 PRINT"■■■■■■■■■■■■■■■■■■LA BOURSE"
```

Dans chaque ligne du programme, le premier caractère placé après les guillemets est un [S] en contraste inversé. Il ramene le curseur à l'origine et <u>facilite</u> ainsi l'affichage sur des coordonnées précices.

Nous verrons plus tard que l'exercice 22 peut s'écrire sur une seule ligne BASIC et économiser des mémoires avec l'emploi de la fonction SPC.

<u>**EXERCICE 23**</u> A la page 21 tu as fait <u>manuellement</u> un rectangle. Cet exercice 23 fera tout ce travail automatiquement... et même davantage! L'opération sera ralentie par des boucles d'attente

```
10 PRINT"■■■■■■■■■■■■";
20 FORI=1 TO 12:PRINT"*";:FOR T=1 TO 150:NEXT T:NEXT I
30 FORI=1TO10:PRINT"*■■";:FOR T=1 TO 150:NEXT:NEXT
40 FOR I=1 TO 12:PRINT"*"+"■■";:FOR T=1 TO 150:NEXT:NEXT
50 FOR I=1 TO 11:PRINT"*■■";:FOR T=1 TO 150:NEXT:NEXT
55 PRINT"■■■■■■ON EST SI"
60 PRINT"■■■■■■■■■MIGNONS"
65 PRINT"■■■■■■■■■FAUT SE"
70 PRINT"■■■■■■■■■PARLER"
75 GOTO 75
```

Il faut que tu parviennes à lire rapidement le message fourni par les caractères de commandes d'affichage inclus entre les guillemets... Par exemple à la ligne 10 tu diras: <u>efface l'écran</u>,... remonte <u>à l'origine</u>,... <u>descends</u> de 5 rangées,... <u>déplace-toi</u> de 4 colonnes vers la droite,... et ne retourne pas à la ligne suivante (pas de retour chariot).

Tu as toutes les notions voulues pour interpréter ce programme. Tu pourras <u>refaire</u> l'exercice <u>en changeant</u> le message, le format, l'encadrement (ex.:Tape Commodore et +)...

L'important est de "pratiquer" afin d'acquérir la

rapidité, les astuces, l'intuition... du programmeur (chasseur) qui suit son curseur "à la trace"!

Comment introduire un SPRITE dans la mémoire de ton C64:

Il y a des gens qui vont te dire qu'il faut être fort en math pour utiliser des SPRITES (pages 8+9) dans tes programmes. Au lieu d'écouter ces "prophètes de malheur", promets-moi de lire attentivement les 4 prochaines pages et de faire sérieusement tous les exercices...En retour tu auras la joie de sortir vainqueur et de pouvoir montrer à tes amis les petites merveilles que tu auras toi-même créées. Ces notions sont divisées en 3 parties. La quatrième sera ton programme et les explications qui l'accompagnent.

1- Notation binaire:

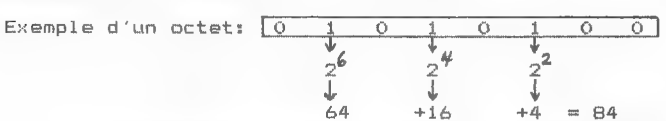
Tu connais bien la notation décimale, c'est celle que tu emploies pour écrire les nombres. Exemple: 84 cents. Décimal veut dire 10...On dit aussi notation à base 10. Ainsi le nombre décimal 84 est formé de 8 dizaines et 4 unités: 8X10 + 4X1 = 84. Bravo!

Le mot binaire te fait penser à deux...Tu as raison car la notation binaire utilise seulement deux symboles: 0 et 1. De plus, la notation binaire est une notation à base 2

Comment obtenir 84 avec une notation à base 2?
a) 2x2x2x2x2x2 = 64 . Comme on a multiplié par 2 six fois, tu peux dire: 2 puissance 6 =64 ou 2^6 = 64 ou 2↑6 = 64
Note: ↑ est le symbole de "puissance" ou exponentiation.
Si tu multiplies par 2 sept fois tu obtiendras 128, c'est-à-dire un nombre supérieur à 84.
b) Ajoute cette valeur: 2x2x2x2 = 16 ou 2^4 = 16 ou 2↑4 = 16.
Si tu multiplies une cinquième fois par 2 tu auras 32...c'est trop car 64+32 = 96.
c) Ajoute cette valeur: 2X2 = 4 ou 2^2 = 4 ou 2↑2 = 4
Ainsi 64+16+4 = 84. Tu as donc trouvé 84 en passant par la notation binaire. Deux fois bravo!

2- Un OCTET (anglais:BYTE):

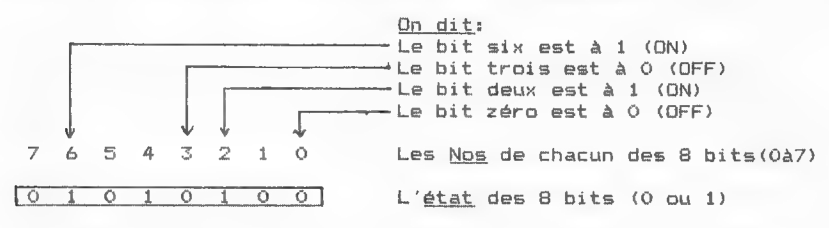
Pour bien comprendre la notation binaire, prenons l'exemple d'un OCTET(ou BYTE). Un octet contient 8 bits ou 8 unités d'information. Chaque bit (binary digit) peut prendre deux états: soit l'état 0 ou l'état 1.



Cet octet est le profil binaire de la lettre T car le code

de ce caractère est 84 (voir tableau p. 76).

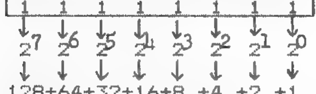
IL FAUT SAVOIR qu'un octet est toujours numéroté de 0 à 7 afin d'identifier chacun des bits de l'octet. (Note: 0à7 et non 1à8 car l'ordinateur inclut le zéro dans ses comptes!)



0 +64+0 +16+0 +4 +0 +0 = 84 Valeur en décimal de l'octet

Voici un octet qui va t'aider dans les prochains exercices car tous ses bits sont à 1 et ainsi il te donne la valeur en décimal de chacun des 8 bits

7 6 5 4 3 2 1 0 Les numéros de chacun des 8 bits



Tous les bits sont à un (ON)

128+64+32+16+8 +4 +2 +1 = 255 Valeur maximale que peut contenir un octet (byte)

EXERCICE 24 Si le bit 7 est à 1 (ON), quel sera sa valeur en décimal? (Réponse après l'exercice 30)

EXERCICE 25 Si le bit 3 est à 0 (OFF), quelle sera sa valeur en décimal?

EXERCICE 26 Quelle est la valeur décimale de l'octet suivant: 00010001?

EXERCICE 27 Ecris la valeur décimale 69 en binaire:

EXERCICE 28 Ecris en mode direct:

PRINT CHR$(84) RETURN

Tu demandes à l'ordinateur d'imprimer le caractère (CHARACTER) No 84. Ton C64 traduit automatiquement la valeur décimale 84 en binaire: "01010010" et affiche le caractère T (Voir tab.p 76). Si tu écris seulement: PRINT 84 ton C64

traduira 84 en binaire et affichera le nombre 84.

EXERCICE 29 L'octet <u>01001110</u> est le profil binaire de quelle lettre?

EXERCICE 30 MESSAGE SECRET! Essaie de découvrir ce message écrit en langage binaire:

```
     01001110    01000001    01000100     01001001     01000101
00100000   01000101   01010011   01010100   00100000   01001010
01001111   01001100   01001001   01000101   00100000   01000011
01001111   01001101   01001101   01000101   00100000   01010101
01001110   00100000   01110011   00101110
```

REPONSES: Question 24: 128. Question 25: 0
Question 26: 17 (16+1=17) Question 27: 01000101
Question 29: Lettre N car N = 78 (Voir tab. p. 76)
Question 30: Nadie est jolie comme un coeur. (=115)

3- **Les OCTETS d'un SPRITE:**

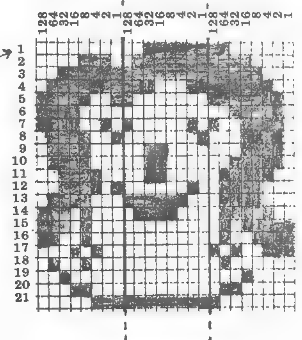
Un sprite contient 63 octets. Pour introduire un sprite dans la mémoire du C64 tu devras **prendre la valeur binaire de chacun des 63 octets** de ton dessin et **traduire cette valeur en décimal.**

Il y a 3 octets par rangée...Tu recueilles leurs valeurs et tu places les résultats dans des 'DATA' **et en ordre**, c'est-à-dire comme dans la lecture d'un texte: de gauche à droite et de haut en bas.

EXERCICE 31 Mets dans des instructions 'DATA', les valeurs décimales des 63 octets de "Miss SPRITE". Il est bon de mettre un max. de 3 rangées (ou 9 octets) par instruction DATA pour faciliter les corrections (ou transform.) **SOLUTION** ci-dessous.

1er GROUPE | 2ième GROUPE | 3ième GROUPE

Rangées



Exemple:

	Groupe 1	Groupe 2	Groupe 3
Rangée 1:	00000000	00111111	11000000
	l'octet = 0 en décimal	l'octet = 63 en décimal	l'octet = 192 en décimal

Première rangée: (du DESSIN ci-dessus)

```
L'octet du groupe 1:                          =0
L'octet du groupe 2: 32+16+8+4+2+1            =63
L'octet du groupe 3: 128+64                   =192
```

Deuxième rangée:

```
L'octet du groupe 1: 8+4+2                    =14
L'octet du groupe 2: 64+32+16+8+4+2+1         =127
L'octet du groupe 3: 128+64+32+16             =240
```

Troisième rangée:

```
L'octet du groupe 1: 32+16+8+4+2+1            =63
L'octet du groupe 2: 128+64+32+16+8+4+2+1     =255
L'octet du groupe 3: 128+64+32+16+8           =248
```

Tu fais ainsi pour les 21 rangées...Ensuite tu mets les valeurs décimales dans des instructions DATA. Fais-le toi-même avant de regarder les réponses ci-dessous:

```
500 DATA 0,63,192,14,127,240,63,255,248
510 DATA 127,227,252,249,128,254,240,0,62
520 DATA 242,130,158,121,1,30,120,48,30
530 DATA 120,48,62,60,48,124,61,2,120
540 DATA 120,252,120,232,120,120,200,0,95
550 DATA 200,0,79,20,0,167,72,0,64
560 DATA 32,0,16,24,0,96,7,255,128
```

Si tu mets 9 valeurs par instruction DATA, tu auras donc 7 lignes BASIC pour les 63 octets de Miss Sprite. **L'instruction DATA sera étudiée au chap. 3.**

Sauve ces DATA de "Miss SPRITE" sur cassette ou sur disque

4- **Miss Sprite à l'écran!:**

EXERCICE 32 Introduis dans ton C64 les 'DATA' de "Miss SPRITE" et ajoute les lignes BASIC Nos 10 à 60:

```
10 FOR N=0 TO 62: READ D: POKE 832+N,D:NEXT N
20 POKE 53269,4:POKE 2042,13
30 POKE 53271,4:POKE 53277,4
40 POKE 53289,0
50 POKE 53252,160
60 POKE 53253,130
500 DATA 0,63,192,14,127,240,63,255,248
510 DATA 127,227,252,249,128,254,240,0,62
520 DATA 242,130,158,121,1,30,120,48,30
530 DATA 120,48,62,60,48,124,61,2,120
540 DATA 120,252,120,232,120,120,200,0,95
550 DATA 200,0,79,20,0,167,72,0,64
560 DATA 32,0,16,24,0,96,7,255,128
```

Avant d'écrire RUN et taper RETURN:

a) Vérifie chacune des instructions et chacune des données contenues dans les DATA (Note: on ne met pas de virgule après la dernière donnée d'une instruction DATA). As-tu 63 valeurs dans les DATA?

b) Pour mieux voir Miss SPRITE, il serait bon d'effacer l'écran et de mettre en blanc le fond de l'écran...Pour obtenir ce résultat ajoute la ligne suivante:

```
5 PRINT CHR$(147):POKE 53281,1
```

Fais exécuter ton programme...:Une jolie "Noirette" apparaît au centre de l'écran.

EXPERIENCES et explications du programme:

La ligne No 10 lit (READ) les DATA et met les 63 valeurs dans les mémoires 832 à 895. (Note: les intructions READ et DATA seront étudiées au chap. 3)

Ligne No 20: Expérience...Remplace le chiffre 4 placé après la virgule par le nombre 251. RUN et RETURN...Le sprite est effacé.

Conclusion: le registre (mémoire) 53269 "allume ou éteint" ton sprite. Remets le chiffre 4 à la place de 251.

L'instruction POKE 2042,13: L'adresse 2042 est le pointeur qui indique à ton C64 qu'il doit aller lire les DATA de ton sprite dans le treizième bloc (832 à 895).

Ligne 30: La première instruction de cette ligne BASIC double la dimension horizontale de ton sprite. La deuxième instruction double la dimension verticale de ton sprite. (Note: Rappelle-toi qu'une ligne BASIC peut contenir plusieurs instructions à condition que celles-ci soient séparées par [:].

Ligne 40: Expérience...Mets 10 au lieu de zéro après la virgule. RUN et RETURN.

Conclusion: le registre 53289 contient la couleur de ton sprite. Notre Miss Sprite devient une jolie "Roussette". (seul son coiffeur le sait!)

Ligne No 50: Expérience...Remplace 160 par 250. RUN et RETURN

Conclusion: le registre 53252 déplace ton sprite horizontalement.

Ligne No 60: Expérience...Remplace 130 par 200. RUN et RETURN.

Conclusion: le registre 53253 déplace ton sprite verticalement.

Une dernière expérience:

Enlève la ligne 30 pour donner à ton sprite ses dimensions normales. Ensuite appuie à la fois RUN/STOP et RESTORE pour que l'écran retrouve son état normal (fond bleu).

Fais exécuter ton programme: RUN et RETURN. Ton sprite retrouve ses dimensions normales (24x21 points).

5- **Miss Sprite se promène:**

EXERCICE 33 Ajoute les lignes 45 et 70 à ton programme

et à la ligne 50, remplace 160 par X:

```
45 FOR X=0 TO 255
50 POKE 53252,X
70 NEXT X:GOTO 45
```

RUN et RETURN. Miss SPRITE se déplace horizontalement.

EXERCICE 34 A la ligne 60, remplace 130 par X.

```
60 POKE 53253,X
```

RUN et RETURN. Miss Sprite se déplace verticalement(ligne 60) et aussi horizontalement(ligne 50). La résultante forme un tracé oblique (de haut en bas vers la droite).

EXERCICE 35 Reprends le dessin du nuage (p. 9) et écris dans des DATA les valeurs décimales des 63 octets (Note: commence au numéro de ligne 2000):

Une solution:

```
2000 DATA 0,0,224,0,1,248,0,59,252
2010 DATA 0,127,254,0,255,255,0,255,255
2020 DATA 1,255,255,7,255,254,15,255,252
2030 DATA 31,255,248,31,255,254,15,255,255
2040 DATA 7,255,255,3,255,254,7,255,248
2050 DATA 31,255,248,63,255,240,127,255,240
2060 DATA 255,247,224,243,227,192,192,192,0
```

Sauve (Save) tes 'DATA' sur cassette ou sur disque. Comme ton nuage annonce le mauvais temps, tu peux l'appeler "NIMBO-STRATUS".

NOTE IMPORTANTE avant de faire l'exercice 36:

Les registres réservés aux SPRITES commencent à l'adresse 53248. Prends l'habitude de mettre 53248 dans une variable **DS** (Début Sprite) et d'initialiser cette variable au commencement de ton programme. Exemple: DS=53248

Tu te sers ensuite de la variable **DS** pour identifier les autres registres de ton programme. Par exemple l'adresse 53269 ou (53248+21) permet "d'allumer" un sprite (Voir exercice 32, ligne 20)...**Tu écriras maintenant: DS+21**

EXERCICE 36 A l'aide des exercices 32 à 34 tu peux construire un petit programme qui déplacera ton nuage (noir) vers le haut sur une trajectoire oblique commençant au milieu de la partie gauche de l'écran

```
5 DS=53248:PRINT CHR$(147):POKE53281,1
10 FOR N=0 TO 62: READ D: POKE 832+N,D:NEXT N
20 POKE DS+21,4:POKE 2042,13
30 POKE DS+23,4:POKE DS+29,4
40 POKE DS+41,11
45 FOR X=6 TO 170
46 FOR T=1 TO 20:NEXT
50 POKE DS+4,X
60 POKE DS+5,170-X
70 NEXT X: GOTO 45
2000 DATA 0,0,224,0,1,248,0,59,252
2010 DATA 0,127,254,0,255,255,0,255,255
2020 DATA 1,255,255,7,255,254,15,255,252
2030 DATA 31,255,248,31,255,254,15,255,255
2040 DATA 7,255,255,3,255,254,7,255,248
2050 DATA 31,255,248,63,255,240,127,255,240
2060 DATA 255,247,224,243,227,192,192,192,0
```

Il y a encore beaucoup de belles choses à apprendre concernant les sprites. Nous ferons de nombreuses expériences **dans le volume deux...**

Entreprends le projet A-2. Il te permettra de mettre en pratique la plupart des notions déjà apprises.



Photo de Miss SPRITE à l'écran, pages 52 et 53.

PROJET A-2

Au cours de l'activité précédente tu as sauvegardé sur cassette les dessins de la maison, de la fusée et des oiseaux et aussi la silhouette du Québec.

a) Introduis le dessin de la maison (projet A-1) dans la mémoire de ton C64.

b) Utilise l'exercice 5 B) pour placer un soleil au-dessus de la maison

c) Utilise ton programme de l'exercice 36 pour que le nuage vienne masquer le soleil, et pendant que le nuage fait disparaître le soleil, fait passer le fond de l'écran du gris clair au gris foncé pour simuler une demi-obscurité.

d) Allume la lampe installée au coin de la maison pour qu'elle éclaire l'escalier. S'il y a des moustiques il serait bon que la lampe soit jaune! Sinon tu peux mettre une ampoule ordinaire (lumière blanche) avec couleur turquoise (anglais: cyan) sur l'escalier.

e) Fais jaillir un éclair entre le nuage chargé d'électricité positive et le paratonnerre.

f) Une demi-seconde après l'éclair tu fais entendre le bruit du tonnerre.

g) Fais scintiller la lampe pendant la décharge électrique et simule un arrêt de courant pendant une demi-seconde.

Ce serait moins drôle si la coupure de courant se faisait au niveau de l'alimentation de ton micro-ordinateur!

h) Fais sortir le SPRITE de l'écran pour avoir progressivement le décor ensoleillé du début

i) Ajoute, au début du programme, les instructions nécessaires pour indiquer quelles touches appuyer pour provoquer: 1-déplacement du nuage, 2-l'éclairage, 3-la foudre et le tonnerre, 4-l'état initial.

NOTE: Tu peux mettre plusieurs instructions par lignes BASIC à condition de les séparer par deux points [:] .

Une suggestion: Tu peux organiser ton programme pour que les opérations précédentes se déroulent soit automatiquement, soit manuellement (clavier).

```
950 PRINT"APPUIE SUR":PRINT"N  POUR NUAGE (SPRITE)
960 PRINT:PRINT"L  POUR LAMPES (ECLAIRAGE)
970 PRINT:PRINT"E  POUR ECLAIR + TONNERRE
980 PRINT:PRINT"D  POUR DEBUT (SOLEIL+MAISON)
985 PRINT"TAPE UNE TOUCHE"
```

(Comment opérer)

```
990 GET A$:IF A$=""THEN 990
```

(Attente d'un caractère quelconque)

```
1000 PRINT""
1005 POKE 53280,15: POKE 53281,1: PRINT CHR$(31)
1010 FOR I=1 TO 5: PRINT: NEXT
1040 PRINT"
1050 PRINT"
1060 PRINT"
1070 PRINT"
1080 PRINT"
1090 PRINT"
1100 PRINT"
1110 PRINT"
1120 PRINT"
1130 PRINT"
1140 PRINT"
1150 PRINT"
1160 PRINT"
1170 PRINT"
1180 PRINT"
1190 PRINT"
```

Le rôle de chacune des lignes de ce programme A-2 est expliqué dans les pages 61 à 63.

(Programme A-1 page 23)

```
1200 E=1024: C=55296
1210 POKE E+93,77:POKE C+93,7: POKE E+95,78:POKE C+95,7
1211 POKE E+132,67:POKE C+132,7:POKE E+134,81:POKE C+134,8
1212 POKE E+136,67:POKE C+136,7: POKE E+173,78:POKE C+173,7
1213 POKE E+175,77:POKE C+175,7
```

(Soleil exercice 5 B p. 39)

```
1225 GET N$: IF N$<>"N"THEN 1225
```

(Attente du caractère N)

```
1227 T2=40: X=11: Z=11
1230 DS=53248
1240 FOR N=0 TO 62:READ D:POKE 832+N,D:NEXT N
1245 RESTORE
1250 POKE DS+21,4:POKE 2042,13
1260 POKE DS+23,4:POKE DS+29,4
1270 POKE DS+41,0
1280 FOR X=Z TO Z+104
1285 FOR T=1 TO T2: NEXT T
1290 POKE DS+4,X
1295 IF X=98 OR X=134 THEN POKE 53281,12: T2=160
1296 IF X=106 THEN POKE 53281,11
1297 IF X=142 THEN POKE 53281,1
1300 POKE DS+5,177-X: IF X=169 THEN 1000
1310 NEXT X
```

(Nuage...SPRITE exercice 36 p.56)

```
1320 GET L$: IF L$<>"L"THEN 1320
```

(Attente du caractère L)

```
1330 POKE C+610,7: POKE C+811,7: POKE C+850,7: POKE C+889,7
1340 FOR I=1 TO 8
1345 IF I>4 THEN POKE C+657+I,1: GOTO 1360
1350 POKE C+621+I,1
1360 NEXT I
```

(Explications p.62)

```
1370 GET E$: IF E$<>"E" THEN 1370              (Attente du caractère E)
1380 POKE E+177,77: POKE C+177,8: POKE E+216,77: POKE C+216,8
1390 FOR I=1 TO 4: POKE C+621+I,14: NEXT
1400 FOR I=1 TO 4: POKE C+661+I,14: NEXT
1410 POKE C+610,14: POKE C+811,14: POKE C+850,14: POKE C+889,14
1420 POKE E+218,77: POKE C+218,8: POKE E+257,99: POKE C+257,8
1430 FOR I=1 TO 4: POKE C+621+I,1: NEXT
1440 FOR I=1 TO 4: POKE C+661+I,1: NEXT
1450 POKE C+610,7: POKE C+811,7: POKE C+850,7: POKE C+889,7
1460 POKE E+258,99: POKE C+258,8: POKE E+259,99: POKE C+259,8
1470 POKE E+260,77: POKE C+260,8: POKE E+261,104: POKE C+261,8
1480 FOR I=1 TO 4: POKE C+621+I,14: NEXT
1490 FOR I=1 TO 4: POKE C+661+I,14: NEXT
1500 POKE C+610,14: POKE C+811,14: POKE C+850,14: POKE C+889,14
1510 FOR V=15 TO 0 STEP -1
1515 POKE 54296,V:POKE 54277,0:POKE 54278,240
1520 POKE 54276,129: POKE 54273,9 : POKE 54272,159
1530 FOR T=1 TO 10:NEXT T:POKE 54273,12:POKE 54272,216
1540 POKE E+177,32:POKE E+216,32:POKE E+218,32:POKE E+257,32
1550 POKE E+258,32:POKE E+259,32:POKE E+260,32:POKE E+261,32
1560 IF V=15 OR V=14 OR V=13 OR V=12 OR V=11 OR V=10 THEN 1600
1570 FOR I=1 TO 4:POKE C+621+I,1:NEXT I
1580 FOR I=1 TO 4:POKE C+661+I,1:NEXT I
1590 POKE C+610,7:POKE C+811,7:POKE C+850,7:POKE C+889,7
1600 NEXT V
1605 POKE S+4,0
1610 GET D$: IF D$<>"D" THEN 1610              (Attente du caractère D)
1620 Z=116: GOTO 1230
2000 DATA 0,0,224,0,1,248,0,59,252
2010 DATA 0,127,254,0,255,255,0,255,255
2020 DATA 1,255,255,7,255,254,15,255,252
2030 DATA 31,255,248,31,255,254,15,255,255
2040 DATA 7,255,255,3,255,254,7,255,240
2050 DATA 31,255,248,63,255,240,127,255,240
2060 DATA 255,247,224,243,227,192,192,192,0
```

(Explications pages 62+63)

QUESTION 1 J'ai introduit dans mon C64 le programe A-2 et après RUN je constate que la lumière jaune n'apparaît pas sur la lampe... quoi faire?

Réponse: Fais les exercices 37 et 38 et tu trouveras probablement la solution de ton problème.

EXERCICE 37 Appuie sur SHIFT et CLR/HOME pour effacer l'écran. Ecris en mode direct, c'est-à-dire sans numéro de ligne, pour ne pas déranger ton programme déjà en mémoire.

POKE 1024,12 RETURN.

Tu viens d'écrire un caractère en haut au coin gauche de l'écran (adresse 1024) mais tu ne le vois pas parce qu'il est écrit en bleu sur fond bleu

Remonte le curseur sur le coin gauche de l'écran et tu verras clignoter la lettre L. Redescends le curseur et ajoute l'instruction suivante, toujours en mode direct:

POKE 55296,1 RETURN. Instantanément la lettre L s'imprimera en blanc sur le coin gauche, en haut de l'écran

Le code écran du caractère L est 12 (voir tableau page 75).
Le code de la couleur blanche est 1 (voir tableau page 38).

EXERCICE 38 Ecris en mode direct:

POKE 1024,81:POKE 55296,7 RETURN. Ta lampe apparaît en jaune au coin de l'écran. (code 81, p. 75) (code 7, p. 38)

Ces expériences te permettent de conclure qu'il y a une case-mémoire pour le caractère et une case-mémoire correspondante pour la couleur.

Réponse à la question 1: Probablement que la case-mémoire de ta lampe et la case-mémoire de la couleur jaune ne coincident pas! A la ligne 1330 de ton programme, tu ramarques que le jaune(7) de la lampe est à l'adresse C+610 (C=55296)...Il faut donc que le caractère lampe(81) soit à E+610 (E=1024)!

Première solution: Voici une méthode rapide pour vérifier si la lampe de ton dessin est bien à E+610:

Ajoute temporairement la ligne BASIC No 1205:

```
1205 POKE E+610,81:POKE C+610,7
```

RUN et RETURN. Tu peux rapidement constater si la couleur

jaune coincide avec la lampe de ta maison...Sinon tu peux changer les adresses des mémoires-écran de la lampe et de l'escalier...Il y en a plusieurs!...Efface ensuite la ligne 1205.

Deuxième solution: Tu peux déplacer l'ensemble de ton dessin soit vers la droite, vers la gauche, vers le bas ou vers le haut pour faire coincider le jaune avec la case E+610

Tu écris: LIST 1000-1190

Ensuite tu amènes le curseur entre le guillemet et le dessin de chaque numéro de ligne

a) Pour déplacer le dessin vers la droite tu appuies **SHIFT et INST** puis RETURN...
b) Pour le déplacer vers la gauche tu appuies **DEL** puis RETURN...
c) Pour descendre le dessin d'une ligne tu ajoutes 1 fois PRINT à la ligne 1010, c'est-à-dire 'TO 7' au lieu de 'TO 6'
d) Pour le remonter d'une ligne tu mets 'TO 5' au lieu de 'TO 6'

En optant pour la deuxième solution tu n'auras aucune correction à faire à l'intérieur de ton programme.

Rôle de chacune des lignes du programme A-2 (pages 58 + 59)

Les lignes 950 à 990:

Présentation sur l'écran des informations nécessaires pour que l'utilisateur puisse savoir quelles touches appuyer et l'effet que chacune d'elle produira. Ce bloc est complété par une boucle d'attente avec GET (voir exercice p.45).

Tu reconnais plusieurs caractères de COMMANDE d'AFFICHAGE (page 35). Ils permettent d'aérer le texte, d'attirer l'attention sur tel ou tel caractère inversé. PRINT permet de laisser une ligne vide. On a donc le même effet qu'avec un caractère de commande d'affichage no 5 page 35).

Ligne 1000: Effacer l'écran...Mettre le cadre gris clair...et le fond de l'écran blanc. Ces registres 53280 et 53281 seront étudiés plus loin.

Ligne 1010: Boucle d'itération qui permet de passer 6 lignes, (PRINT), pour loger le dessin du soleil.

Les lignes 1040 à 1190 forment le dessin de la maison.

Les lignes 1225, 1320, 1370 et 1610:

Ces 4 lignes Basic jouent un rôle identique. Prenons

comme exemple la ligne 1225. Le C64 attend que tu appuies la touche N pour continuer l'éxécution du programme.

Cette attente est provoquée par l'instruction GET suivie de l'instruction IF... THEN (voir GET page 45...). Si tu appuies une touche qui est différente de (<>) N, le C64 revient à la ligne 1225 et continue de "scruter" le clavier... Il est retenu dans une boucle de scrutation et il en sortira lorsque tu taperas la lettre N.

Aussitôt que la variable chaîne N$ contient le caractère N, le programme continue à la ligne 1227 dont le rôle est d'initialiser les variables: a) T2=durée; b) X=position de départ; c) Z=variable de service.

EXERCICE 39

```
1225 GET N$
1226 IF N$="N" THEN 1230
1227 GOTO 1225
1230 PRINT "OK"                    RUN et RETURN
```

Le programme continue seulement si tu appuies sur N. Ces 3 lignes (1225,1226,1227) jouent donc le même rôle que l'unique ligne 1225 de ton programme. Quand tu ne peux utiliser l'opérateur relationnel "différent de" (<>) il faut une instruction GOTO pour revenir à GET (voir exercice 15 p. 46).

Dans le programme A-2 il était chaque fois possible et avantageux (gain de mémoires) d'utiliser l'opérateur <> . C'est une habitude à prendre: relire ton programme et le rendre aussi court que possible.

Lignes 1230 à 1310: Ces lignes correspondent au SPRITE (Nuage) de l'exercice 36

L'instruction RESTORE (ligne 1245) remet le pointeur au début des DATA...RESTORE sera étudié au chapitre suivant avec READ et DATA

Ligne 1330: Quand tu appuies L (à 1320) la ligne 1330 allume la lampe en jaune et projette un éclairage jaune sur les trois marches (voir page 39 pour les mémoires d'écran des CODES-COULEURS et page 38 pour les CODES 0 à 15 des couleurs).

Lignes 1340 à 1360: Eclairage de la fenêtre

Les lignes 1380 à 1605: L'originalité de ce projet A-2 se situe au niveau de ces lignes où 4 opérations se chevauchent pour donner l'illusion de simultanéité:

1- formation d'un éclair (orange) dans le ciel: 1420,1460 et 1470

2- l'éclairage de la maison disparaît: 1390, 1400 et 1410 ainsi que 1480, 1490 et 1500. L'éclairage réapparaît: 1430, 1440 et 1450 ainsi que 1570, 1580 et 1590.

3- le bruit du tonnerre: 1510,1520,1530 et 1605. Ce n'est pas encore du "tonnerre" comme bruit... Tu pourras faire mieux au chapitre suivant quand tu auras étudié le son et le bruit

4- effacer l'éclair: 1540 et 1550

Ligne 1560: Pendant que le volume (V) du bruit passe de 15 à 10, le programme n'exécute pas les lignes 1570 à 1590...ainsi la maison demeure plongée dans l'obscurité environ 1/2 seconde.

Ligne 1620: La variable de service Z=116 permet au nuage de se déplacer (1230...) à partir de sa position sur le soleil.

NOTE Tu remarques que le bruit vient après l'éclair et continue après la disparition de l'èclair.

C'est normal! (Vitesse de la lumière: 300 000 000 mètres/seconde. Vitesse du son environ 340 mètres/seconde.

Première transformation du programme A-2

Sauve d'abord ta version originale sur cassette.

Tu peux rendre automatiques, i.e. sans toucher au clavier, les quatre opérations: nuage, lampe allumée, éclair + tonnerre, retour à l'état initial.

Il suffit de changer les instructions GET et IF par une boucle de délai (d'attente).

EXERCICE 40 Fais cette première transformation
Remplace la ligne 1225 par:

```
1225 FOR T=1 TO 1000:NEXT T
```

Cette boucle dure environ une seconde. Tu remarques alors que ton micro-ordinateur peut compter jusqu'a mille en 1 seconde! (en langage évolué).

Si tu veux que l'attente dure 2 secondes, remplace 1000 par 2000.

Procède de la même façon avec les lignes 1320, 1370 et 1610.

Tu écris RUN et tu te croises les bras!

Deuxième transformation du programme A-2

Mets une ampoule ordinaire (lumière blanche) et éclairage sur l'escalier (couleur turquoise).

Il suffit d'amener le curseur sur le chiffre 7 qui se trouve après la virgule dans les instructions POKE et remplacer le 7 par un autre chiffre (voir le tableau des codes-couleurs d'écran page 38).

EXERCICE 41 Fais cette deuxième transformation à la ligne 1330:

```
1330 POKE C+610,1:POKE C+811,3:POKE C+850,3:POKE C+889,3
```

C'est l'adresse écran du code des couleurs pour la lampe. (C+610)

Ce sont les 3 adresses écran du code des couleurs pour les marches. (C+811, C+850, C+889)

N'OUBLIE PAS de taper RETURN pour que tes corrections passent en mémoire.

Fais le même changement aux lignes 1450 et 1590 (là où la lampe se rallume quand elle clignote).

PROJET B-2 Départ de ta fusée

Introduis le dessin de ta fusée (projet B-1 page 32) dans ton micro-ordinateur.

La fusée est sur l'écran et on attend l'heure "H" du départ.

a) Introduis l'heure réelle sur l'horloge du C64 (voir page 66)

b) Fais imprimer l'heure en contraste inversé en haut de la fusée sur le coin gauche de l'écran (heures, minutes, sec.).

c) Cette horloge devra t'indiquer l'heure continuellement (pour chaque seconde).

d) Utilise l'instruction POKE pour allumer les moteurs et produire un vrombissement semblable à celui que tu as déjà entendu à la TV.

e) Fais partir la fusée pour qu'elle monte et disparaisse en haut de l'écran (calcule environ 2 secondes pour sa montée sur l'écran).

f) N'oublie pas de faire disparaître aussi le vrombissement... car la fusée est déjà très loin!

g) Organise ton programme pour que le départ se fasse soit manuellement (clavier) ou automatiquement (horloge).

Une solution de ce projet B-2 est proposée à la page 65. Avant de commencer ton projet, fais les exercices 42 à 45

PROGRAMME B-2: DEPART de ta FUSEE
Mise à feu <u>manuel</u> (clavier)

```
990 PRINT"▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓"
1000 PRINT"                    ♠
1010 PRINT"                   /‾\
1020 PRINT"                   | |
1030 PRINT"                   | |
1040 PRINT"                   | |
1050 PRINT"                  |A|
1060 PRINT"                  |R|
1070 PRINT"                  |T|
1080 PRINT"                  |H|
1090 PRINT"                 / A \
1100 PRINT"                 | B |
1110 PRINT"                 | A |
1120 PRINT"                 | S |
1130 PRINT"                 | K |
1140 PRINT"                 | A |
1150 PRINT"                /|   |\
1160 PRINT"                || I ||
1170 PRINT"                ||   ||
1180 PRINT"                ||___||
1190 PRINT"                V/\ /\V
1200 PRINT"                 ▒▒ ▒▒
1300 PRINT"▓▓" TI$ SPC(20) "TAPE D▓";
1320 GET D$: IF D$="D" THEN 1340
1330 GOTO 1300
1340 FOR F=1 TO 20
1350 POKE 54296,15: POKE 54277,0: POKE 54278,240: POKE 54276,129
1351 POKE 54273,12 : POKE 54272,216
1360 C=55296: POKE C+937,7: POKE C+938,2: POKE C+940,7: POKE C+941,2
1370 POKE C+937,2: POKE C+938,7: POKE C+940,2: POKE C+941,7
1380 NEXT F
1385 FOR B=1 TO 12:PRINT"▓":NEXT
1390 FOR H=1 TO 25
1400 PRINT
1410 FOR T=1 TO 20: NEXT T
1420 NEXT H
1430 FOR V=15 TO 0 STEP-.05: POKE 54296,V: NEXT V: POKE 54276,0
1440 GOTO 990
```

Le rôle de chacune des lignes est expliqué: pages 67 et 68
Le caractère <u>damier</u> (pour les moteurs) s'obtient avec les touches Commodore et +

PROGRAMME B-3: DEPART <u>automatique</u> (horloge) de ta FUSEE

Il suffit de changer la ligne 1320. Enlève aussi "TAPE D" dans 1300 et tu auras ceci:

```
1300 PRINT"▓▓" TI$ "▓▓▓▓▓▓▓";
1320 IF TI$="203050" THEN 1340
```

Si tu veux que le départ ait lieu à 16 hres, remplace "203050" par "160000". (mets d'abord ton C64 à l'heure de ta montre).

L'HORLOGE du C64

Les variables TI et TI$ sont en relation avec le <u>temps réel</u> d'une minuterie incluse dans le C64, c'est pourquoi on ne peut utiliser ces variables à d'autres fins dans un programme.

TI et TI$ partent à 0 seconde quand le C64 est mis à "ON" et sont mises à l'heure à tous les 1/60 seconde.

EXERCICE 42 **a) Ecris** en mode direct

PRINT TI

Si ton ordinateur est à "ON" depuis 10 secondes tu liras 600 (après avoir appuyé sur RETURN) parce que 600 fois 1/60 seconde = 10 secondes.

b) Ecris: PRINT TI/60 RETURN.
Cette fois tu obtiens des secondes.

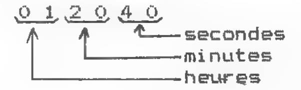
c) Ecris: PRINT TI/60/60 RETURN
Tu auras des minutes et TI/60/60/60 te donnera des heures.

TI$ est une variable chaîne de caractères qui t'indique les heures, les minutes et les secondes sous la forme d'une chaîne de 6 chiffres.

EXERCICE 43 Ecris en mode direct

PRINT TI$ RETURN

Si ton C64 est à "ON" depuis 1 heure 20 minutes 40 secondes tu auras sur l'écran:



TI$ permet d'ajuster l'heure du C64 à l'heure de ta montre

EXERCICE 44 Supposons que ta montre indique 20h 15m 30s. Ecris

TI$="201530" RETURN

Maintenant le C64 te donnera l'heure exacte chaque fois que **tu écriras: PRINT TI$** RETURN
(soit en mode direct, soit dans un programme)

Cette horloge (ou minuterie) du C64 te sera utile dans un grand nombre d'applications pratiques... Elle a les limites de ton imagination!

Cette fois-ci cette horloge va te servir pour le départ de ta fusée et pour t'indiquer l'heure au coin gauche de l'écran (projet B-2 page 64).

EXERCICE 45 Ecris l'heure exacte, en contraste inversé, au centre de l'écran. Cette horloge devra indiquer l'heure à chaque seconde.

```
5 PRINT"□"
10 M$="■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"  ← RVSon, HOME, 11 CRSR bas
20 PRINT M$ TI$ "■■■■■■■";: GOTO 20       16 CRSR droite: (page 35)
```

QUESTION 2 J'ai introduit dans le C64 le programme B-2 de la page 65 et je constate que la fusée s'élève mais les moteurs ne s'allument pas! Que se passe-t-il?

Réponse: Il faut que les mémoires d'écran des CODES-COULEURS coincident avec les mémoires d'écran des CODES-CARACTERES (voir pages 37 et 39). Le programme de la page 65 anime une fusée dont la tête est située à la colonne 19 (n'oublie pas de compter la colonne 0).

Regarde le modèle tracé sur une feuille quadrillée (page 70).

Compare le dessin de ta fusée avec celui de la page 70 et s'il est trop à gauche déplace-le vers la droite de une ou deux ou... colonnes selon la méthode indiquée à la 2ième solution de la page 61.

Si ton dessin est trop à droite, il faut appuyer INST/DEL sans SHIFT.

Rôle de chacune des lignes du programme B-2

Ligne 900: Le premier caractère de COMMANDE d'AFFICHAGE vide l'écran et les 30 autres font descendre le curseur au delà de la 25e ligne (no 5 page 35) afin de donner l'impression que la fusée sort du bas de l'écran.

Lignes 1000 à 1200 forment le dessin de la fusée.

Ligne 1300: Cette ligne **commande 7 choses au 'curseur' du C64: 1- Imprime en contraste inversé (No 7 p.35), 2- à partir de l'origine (No 2 p.35), 3- l'heure (TI$), 4- fais 28 espaces [SPC (28)], 5- imprime 'TAPE D', 6- remonte d'une ligne (No 6 p.35), 7- et ne fais pas de 'retour' chariot (;). Note: La fonction SPC sera étudiée au chapitre suivant.**

Quand ton C64 a imprimé 'TAPE D', le caractère D étant sur la dernière colonne, le curseur est passé au début de la rangée suivante. Il fallait donc le remonter d'une rangée pour que l'heure s'imprime toujours au même endroit de l'écran.

Ligne 1320: L'ordinateur demande le caractère D. Si tu

appuies D le programme continue à la ligne 1340.

Ligne 1330: Si le caractère D n'a pas été tapé le C64 redemande ce caractère mais en passant par la ligne 1300 pour que l'horloge continue de fonctionner sur l'écran.

Ligne 1340: Début d'une boucle d'itération.

Ligne 1350: L'adresse 54296 est celle du volume... (comme le bouton volume de la TV...: fort ou faible). Le code 15 indique que le volume est au maximum...Tu étudieras les registres du son et du bruit au chapitre suivant

Ligne 1360: Simulation de la Mise à feu des moteurs avec le jaune (7) et le rouge (2).

Ligne 1370: Permutation des couleurs rouge et jaune pour donner l'illusion du feu.

Ligne 1380: La variable F est incrémentée de 1, ensuite retour à 1340 pour une 2e itération... et ainsi de suite jusqu'à ce que F = 20. Cette boucle dure environ 3 secondes pour permettre d'observer le phénomène de la mise à feu.

Linge 1385: Cette boucle amène le curseur sur la dernière rangée de l'écran.

Lignes 1390 à 1420: Le curseur étant sur la dernière rangée, chaque itération provoque un 'retour chariot' (PRINT) jusqu'à ce que la fusée soit complètement disparue de l'écran

La ligne 1410 est une boucle d'attente (imbriquée) d'environ 2/100 de seconde. Cette attente se répète à chaque itération... ce qui permet à la fusée de parcourir l'écran en deux secondes environ...

Ligne 1430: Le bruit diminue d'intensité jusqu'à ce que le volume V = 0

Ligne 1440: Retour au début du programme.

ORDINOGRAMME des lignes 1310 à 1340

L'heure exacte est toujours indiquée, même si le C64 "attend" le caractère D.

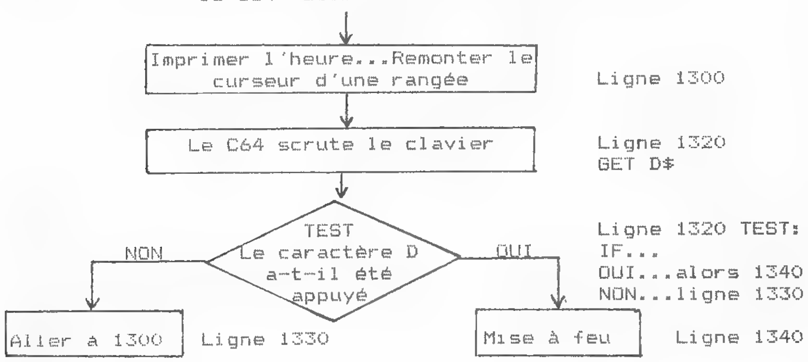
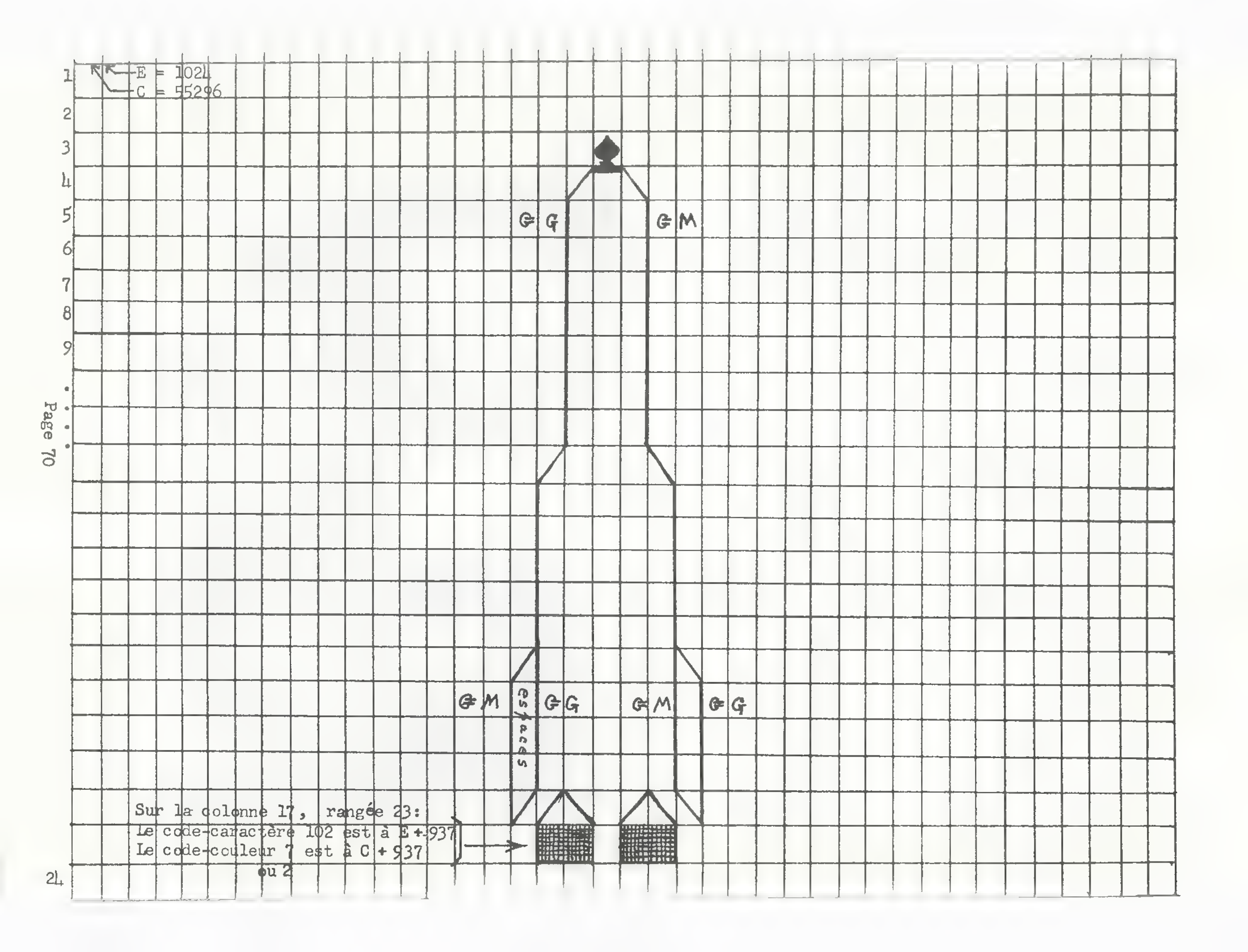




Photo du projet B-2, pages 64 et 65



E = 1024
C = 55296
1
2
3
4
5
6
7
8
9
24
G G
G M
G M
espaces
G G
G M
G G
Sur la colonne 17, rangée 23:
Le code-caractère 102 est à E + 937
Le code-couleur 7 est à C + 937
ou 2

PROGRAMME B-4: FUSEE <u>en couleur</u> (avec un observateur!)

```
990 PRINT""
1000 PRINT"
1010 PRINT"
1020 PRINT"
1030 PRINT"
1040 PRINT"
1050 PRINT"                    A
1060 PRINT"                    R
1070 PRINT"                    T
1080 PRINT"                    H
1090 PRINT"                    A
1100 PRINT"                    B
1110 PRINT"                    A
1120 PRINT"                    S
1130 PRINT"                    K
1140 PRINT"                    A
1150 PRINT"
1160 PRINT"                    I
1170 PRINT"
1180 PRINT"
1190 PRINT"
1200 PRINT"
1300 PRINT"" TI$ SPC(28) "TAPE D";
1320 GETD$:IFD$="D"THEN1340
1330 GOTO 1300
1340 FOR F=1 TO 20
1350 POKE 54296,15: POKE 54277,0: POKE 54278,240: POKE 54276,129
1351 POKE 54273,12: POKE 54272,216
1360 C=55296: POKE C+937,7: POKE C+938,2: POKE C+940,7: POKE C+941,2
1370 FOR T=1 TO 10:NEXT:POKE C+937,2:POKE C+938,7:POKE C+940,2:POKE C+941,7
1380 NEXT F
1385 FOR B=1 TO 12: PRINT"": NEXT
1390 FOR H=1 TO 25
1400 PRINT
1410 FOR T=1 TO 20: NEXT T
1420 NEXT H
1430 FOR V=15 TO 2 STEP-.05: POKE 54296,V: NEXT V
1440 PRINT""
1450 PRINT"
1460 PRINT"
1461 PRINT"
1470 PRINT"QUE VOIS-TU?"
1480 X=TI/60
1490 IF TI/60-X<6 THEN 1490
1500 POKE 53280,6: POKE 53281,6
1510 PRINT"JE VOIS UN POINT ROUGE":PRINT"MAIS CA BOUGE TROP!"
1520 FOR I=1 TO 20
1525 POKE 53265,28: FOR T=1 TO 50: NEXT
1530 E=1024: POKE E+459,46: POKE E+461,46: POKE E+500,42: POKE E+539,39
1535 POKE E+541,39: POKE C+459,7: POKE C+461,7: POKE C+500,2
1540 POKE C+539,7: POKE C+541,7: POKE 53265,27: FOR T=1 TO 50: NEXT
1550 POKE 53265,26: FOR T=1 TO 50: NEXT
1560 POKE 53265,27: FOR T=1 TO 50: NEXT
1570 NEXT I: POKE 54296,0
1580 GOTO 990
```

Pour obtenir le programme B-4 de la page précédente, tu "charges" ton programme B-2 dans le C64 et tu introduis les couleurs dans ton dessin au moyen des caractères de COMMANDE d'AFFICHAGE (nos 7 et 8 page 35) et des caractères de COMMANDE de COULEURS (voir **tableau page 74**). Ce tableau sera étudié dès le début du prochain chapitre.

Tu ajoutes ensuite les lignes 1440 à 1580.

Tu remarqueras l'emploi de l'horloge pour obtenir une boucle d'attente de 6 secondes: lignes 1480 et 1490.

La mémoire 53265 permet de déplacer verticalement le fond de l'écran. Exemple: POKE 53265,27 (position normale)
POKE 53265,26 (fait monter l'écran légèrement, 25 davantage...
POKE 53265,28 (fait descendre légèrement, 29 davantage...).

Je n'ajoute pas d'autres explications pour ce programme B-4. L'effort que tu fourniras pour le comprendre te permettra de résumer les connaissances acquises en langage BASIC.

CONT

La commande CONT dit au C64 de continuer l'exécution d'un programme que tu as arrêté avec la touche STOP. CONT fera aussi continuer l'exécution d'un programme qui a été arrêté par une instruction STOP ou END inscrite dans le programme (voir exercice 16 du chapitre IV).

Expérience Lorsque ta fusée monte sur l'écran appuie la touche STOP et écris

CONT puis tape RETURN

NOTE 1- Cette commande ne fonctionne pas lorsque tu arrêtes le déroulement du listing d'un programme.

2- On utilise généralement la commande CONT en liaison avec une instruction STOP introduite temporairement dans un programme pour la recherche des erreurs...

CHAPITRE III

ACTIVITES DIRIGEES Série No 3

BUTS Cette troisième série d'activités a pour but de te familiariser avec

1- Les 16 caractères de COMMANDE des COULEURS

2- La notion de compteur

3- Les 16 couleurs du fond et du cadre de l'écran

4- Une technique: la balançoire

5- Les opérateurs logiques

6- Etude de six fonctions: FRE, SPC, TAB, ASC, CHR$, PEEK

7- Le tableau des caractères ASCII

8- Quatre nouvelles instructions: READ, DATA, RESTORE, et REM

9- La programmation du son et de la musique.

Les 16 caractères de <u>COMMANDE</u> des <u>COULEURS</u>

Ces caractères apparaissent lorsque tu **ouvres les guillemets** et que tu tapes les touches suivantes: (en mode direct ou prog.)

Nos	Appuyer simultanément sur les touches:	Caractère de commande de couleur:	Le rôle de chacun des caractères:	Codes ASCII (voir p.95) CHR$(↓)
1-	CTRL 1	■	noir	144
2-	CTRL 2	E	blanc	5
3-	CTRL 3	£	rouge	28
4-	CTRL 4	◥	turquoise	159
5-	CTRL 5	▚	pourpre	156
6-	CTRL 6	↑	vert	30
7-	CTRL 7	←	bleu	31
8-	CTRL 8	π	jaune	158
9-	C= 1	♠	orange	
10-	C= 2	╭	brun	
11-	C= 3	╳	rouge clair	
12-	C= 4	○	gris foncé	
13-	C= 5	♣	gris perle	
14-	C= 6	▐	vert clair	
15-	C= 7	◆	bleu clair	
16-	C= 8	┼	gris clair	

Tableau des CODES-CARACTERES d'ECRAN

Avec POKE			POKE		POKE		POKE			POKE			POKE	
0	@		27	[	48	Ø	64			91			106	
1	A	a	28	£	49	1	65	♠	A	92			107	
2	B	b	29	]	50	2	66		B	93			108	
3	C	c	30	↑	51	3	67		C	94	π		109	
4	D	d	31	←	52	4	68		D	95			110	
5	E	e	32	ESPACE	53	5	69		E	96	ESPACE		111	
6	F	f	33	!	54	6	70		F	97			112	
7	G	g	34	"	55	7	71		G	98			113	
8	H	h	35	#	56	8	72		H	99			114	
9	I	i	36	$	57	9	73		I	100			115	
10	J	j	37	%	58	:	74		J	101			116	
11	K	k	38	&	59	;	75		K	102			117	
12	L	l	39	'	60	<	76		L	103			118	
13	M	m	40	(	61	=	77		M	104			119	
14	N	n	41	)	62	>	78		N	105			120	
15	O	o	42	*	63	?	79		O				121	
16	P	p	43	+			80		P				122	
17	Q	q	44	,			81	●	Q				123	
18	R	r	45	—			82		R				124	
19	S	s	46	.			83	♥	S				125	
20	T	t	47	/			84		T				126	
21	U	u					85		U				127	
22	V	v					86		V					
23	W	w					87	○	W					
24	X	x					88	♣	X					
25	Y	y					89		Y					
26	Z	z					90	♦	Z					

↑ Mode texte (colonne des minuscules)
↑ Mode graphique (normal) (colonne des majuscules)

NOTE: Pour obtenir le caractère en **contraste inversé** il suffit **d'ajouter 128** au code ci-dessus.

TABLEAU des CODES ASCII du C64 (en décimal)

PRINT CHR$( ) — CHR$( ) — PRINT ASC("")

0		28	RED	56	8	84	T	110		138	f4	166	
1		29	CRSR →	57	9	85	U	111		139	f6	167	
2		30	GRN	58	:	86	V	112		140	f8	168	
3		31	BLU	59	;	87	W	113	●	141	SHIFT RETURN	169	
4		32	SPACE	60	<	88	X	114		142	SWITCH TO UPPER CASE	170	
5	WHT	33	!	61	=	89	Y	115	♥	143		171	
6		34	"	62	>	90	Z	116		144	BLK	172	
7		35	#	63	?	91	[	117		145	CRSR ↑	173	
8		36	$	64	@	92	£	118		146	RVS OFF	174	
9		37	%	65	A	93	]	119	○	147	CLR HOME	175	
10		38	&	66	B	94	↑	120	♣	148	INST DEL	176	
11		39	'	67	C	95	←	121		149		177	
12		40	(	68	D	96		122	♦	150		178	
13	RETURN	41	)	69	E	97	♠	123		151		179	
14	SWITCH TO LOWER CASE	42	*	70	F	98		124		152		180	
15		43	+	71	G	99		125		153		181	
16		44	,	72	H	100		126	π	154		182	
17	CRSR ↓	45	-	73	I	101		127		155		183	
18	RVS ON	46	.	74	J	102		128		156	PUR	184	
19	CLR HOME	47	/	75	K	103		129		157	CRSR ←	185	
20	INST DEL	48	Ø	76	L	104		130		158	YEL	186	
21		49	1	77	M	105		131		159	CYN	187	
22		50	2	78	N	106		132		160	SPACE	188	
23		51	3	79	O	107		133	f1	161		189	
24		52	4	80	P	108		134	f3	162		190	
25		53	5	81	Q	109		135	f5	163		191	
26		54	6	82	R			136	f7	164			
27		55	7	83	S			137	f2	165			

Les 16 caractères de COMMANDE de COULEUR

Tu es sans doute familier présentement avec les 11 caractères de commande d'affichage.

Il faut maintenant que tu deviennes aussi habile avec les 16 caractères de COMMANDE de COULEUR. Le tableau de la page 74 t'indique le symbole et le rôle que joue chacun de ces caractères.

EXERCICE 1-A Utilise la couleur pourpre et écris BONNE FETE en mode direct:

```
PRINT"▒BONNE FETE
```

Sur la partie verticale de la touche [–] il y a 2 caractères graphiques; celui de gauche ▒ est utilisé, en contraste inversé, pour symboliser la couleur rouge. C'est ce caractère que tu vois entre les guillemets au début de la chaîne (No 5 p. 74).

EXERCICE 1-B Refais l'exerc. 1-B en contraste inversé et commande un curseur bleu foncé: FOND BLANC POUR VOIR LES COULEURS

```
POKE 53281,1:PRINT"▒▒BONNE FETE▒▒"
```

La touche [←] est située en haut du clavier, à gauche. C'est ce caractère, en contraste inversé qui sert de symbole pour la couleur bleu foncé et que tu vois après le mot FETE (No 7 p. 74)

EXERCICE 2 Ecris ce petit programme:

```
5 POKE 53281,1
10 PRINT"▒HAUTE TENSION▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";
20 FOR T=1 TO 1000: NEXT
30 PRINT"▒   DANGER▒    ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";
40 FOR T=1 TO 1000: NEXT
50 GOTO 10
```

Dans ce programme tu as utilisé 2 nouveaux caractères de commande de couleur pour obtenir le jaune et le noir (voir les nos 8 et 1 du tableau page 74).

Sur la partie verticale de la touche [↑] tu vois le caractère "pi" [π]. C'est ce caractère π en contraste inversé qui sert de symbole pour la couleur jaune.

Le symbole de la couleur noire est le caractère espace en contraste inversé avec une rangée éteinte en haut et une colonne éteinte à droite (bits à 0). Sur l'écran il a donc la forme d'un rectangle un peu plus petit que le curseur: ▮

Le caractère de commande de la couleur jaune est au début de la chaîne de caractères de la ligne 10 et celui de la couleur noire est au début de la chaîne de caractères de la

ligne 30.

N'oublie pas que les deux chaînes doivent contenir le même nombre de caractères pour qu'il y ait superposition des 2 messages sans laisser de trace du premier! Tu comprends alors le rôle des caractères espaces dans la chaîne de la ligne 30.

EXERCICE 3 Ecris un court programme qui fait clignoter les mots GRANDE VENTE en contraste inversé et de la couleur rouge. Organise le programme pour que le message apparaisse pendant 1 seconde et disparaisse pendant 1/2 seconde.

```
10 PRINT"♥": POKE 53281,1
20 FOR T=1 TO 500: NEXT
30 PRINT"▓▓GRANDE VENTE█"
40 FOR T=1 TO 1000: NEXT
50 GOTO 10
```

Dans ce programme c'est tout l'écran qui s'efface et demeure dans cet état pendant 1/2 seconde. Le symbole de la couleur rouge est le premier caractère à l'intérieur des guillemets de la ligne no 30.

En amérique notre unité monétaire est le dollar: [$]. En Angleterre l'unité monétaire principale est la Livre stirling: £ . Sur le clavier du C64 la touche [£] est située à gauche de la touche CLR/HOME. C'est le caractère: £ en contraste inversé qui sert de symbole pour la couleur rouge.

REMARQUE: Quand tu tapes sur le clavier du C64 et que tu veux que l'écran affiche les caractères dans une couleur plus lisible ou plus "reposante"...,tu appuies à la fois sur CTRL et le chiffre ayant la couleur qui te plaît!

EXERCICE 4 Dessine une petite auto dont les contours seront verts et allume les phares pour qu'ils éclairent (en blanc) sur une distance de 2 colonnes. Mets les roues de couleur turquoise.

```
5 PRINT"♥": POKE 53281,0
10 PRINT" ▓╱‾‾‾╲    ▓◤"
20 PRINT" |    ‾╲▓◤ ▓"
30 PRINT" |_____▓▓▓  "
40 PRINT"◣  O  O ▓▓▓ "
50 PRINT"______________________"
```

Le C64 se charge de fermer les guillemets de façon fictive car ils n'ont été ouverts qu'une fois par ligne BASIC.

La touche [↑] en contraste inversé forme le symbole de la couleur verte. Le symbole du blanc est le caractère [E] en contraste inversé. Le symbole de la couleur turquoise (anglais: Cyan) est le triangle [◣] qui forme le caractère graphique inversé sur la gauche de la touche étoile (voir les nos 2, 4, 6 du tableau page 74).

Une conclusion importante: Tu viens d'apprendre que si les **guillemets sont ouverts** et que tu appuies **CTRL et l'un des 8 chiffres**, un symbole spécial apparaît et ce symbole correspond à l'une des couleurs inscrites sur la partie verticale des touches des chiffres: BLK(noir)......YEL(jaune).

Ce n'est pas tout! Le C64 a 8 autres couleurs...Si les guillemets sont ouverts et que tu appuies la **touche COMMODORE** et l'un des 8 chiffres, il apparaîtra aussi un symbole spécial:

```
1 ORANGE        Symbole:  Shift A inversé  (Voir p.74 No  9)
2 BRUN          Symbole:  Shift U inversé  (Voir p.74 No 10)
3 ROUGE CLAIR   Symbole:  Shift V inversé  (Voir p.74 No 11)
4 GRIS FONCE    Symbole:  Shift W inversé  (Voir p.74 No 12)
5 GRIS PERLE    Symbole:  Shift X inversé  (Voir p.74 No 13)
6 VERT CLAIR    Symbole:  Shift H inversé  (Voir p.74 No 14)
7 BLEU CLAIR    Symbole:  Shift Z inversé  (Voir p.74 No 15)
8 GRIS CLAIR    Symbole:  Shift + inversé  (Voir p.74 No 16)
```

EXERCICE 5-A Introduis dans un DATA les 8 couleurs obtenues avec CTRL...Ensuite dans un 2ième DATA les 8 couleurs obtenues avec COMMODORE...**(N'oublie pas de placer ces symboles entre guillemets):**

```
100 DATA "■","■","■","◣","■","■","■","■"
110 DATA "■","■","■","■","■","■","■","■"
```

NOTE: Ces caractères de COMMANDE → soit de COULEUR (p. 74) / → soit d'AFFICHAGE (p. 35)
n'apparaissent pas lors de l'exécution du programme, c'est-à-dire quand tu écris RUN et tapes RETURN...Mais le caractère qui suit un caractère de COMMANDE s'imprime dans la case précédente, là où se trouve le caractère de COMMANDE dans les guillemets.

EXERCICE 5-B Les caractères deviennent **illisibles** avec certaines couleurs formant le fond de l'écran! Ce programme écrit le mot MARIE en 16 couleurs différentes sur 4 sections d'écran. Le **fond de l'écran** peut changer 16 fois (une nouvelle couleur chaque fois) en tapant une seule touche:

```
5 FOR CF=0 TO 15
10 PRINT"■": POKE 53281,CF
15 PRINT"A TOI DE CHOISIR■■■"
16 PRINT"COULEUR DU FOND: NO " CF "(PAGE 38)■■■"
20 FOR J=1 TO 16
25 READ C$: PRINT C$;J"MARIE",
30 IF J=4 OR J=8 OR J=12 THEN PRINT
35 NEXT J: RESTORE
40 PRINT "■■■TAPE ■F"
45 GET A$: IF A$<>"F" THEN 45
50 NEXT CF: PRINT"■■": POKE 53281,6
100 DATA "■","■","■","◣","■","■","■","■"
110 DATA "■","■","■","■","■","■","■","■"
```

Tu remarques que les numéros des couleurs commencent à 0 avec POKE (noir) et à 1 avec PRINT (noir).

REM

Les instructions REM ne sont pas exécutées quand tu écris RUN et appuies RETURN.

REM (REMarque) permet d'ajouter des commentaires dans le listing d'un programme pour en faciliter la compréhension. Exemple: le but du programme; le rôle d'une variable, d'une formule, d'un sous-programme, etc...

REM peut être placé au début d'une ligne BASIC ou plus loin dans la ligne à condition qu'il soit précédé de : . Exemple:

```
5 REM Ce programme calcule la racine carrée des nombres 1 à 100
10 REM  Les résultats seront arrondis à deux décimales.
15 N=100*3*2:REM Le caractère * est le symbole de la multiplic.
20......
```

Le C64 commencera l'exécution du programme à la ligne 15...

Notion de COMPTEUR

Dans un programme, un compteur est une variable de contrôle qui compte le nombre de fois qu'une ou des instructions seront exécutées.

EXERCICE 6 Ecris un programme qui affiche sur l'écran le carré des 20 premiers nombres.

```
10 PRINT"NOMBRES","AU CARRE"
20 I=1
30 PRINT I,
40 C=I*I
50 PRINT C
60 I=I+1
70 IF I>20 THEN END
80 GOTO 30
```

Ligne 20: Initialisation du compteur (obligatoire ici parce que I commence avec une valeur différente de 0).

Ligne 60: Incrémentation de 1 du compteur. Note: Pour obtenir le carré des nombres impairs seulement, il faudrait incrémenter le compteur de 2: (I=I+2). Fais l'expérience...

Ligne 70: Test d'arrêt.

Tu peux obtenir le même résultat avec une boucle FOR... NEXT au lieu du compteur.

EXERCICE 7 Remplace la ligne 20 de l'exercice 6 par:

```
20 FOR I=1 TO 20
```

Efface les lignes 60 et 70 et remplace la ligne 80 par:

```
80 NEXT I
```

Tu auras:

```
10 PRINT"NOMBRES","AU CARRE"
20 FOR I=1 TO 20
30 PRINT I,
40 C=I*I
50 PRINT C
80 NEXT I
```

Ligne 20: FOR I=1 TO 20 signifie: exécute toutes les instructions qui suivent jusqu'à l'instruction NEXT, et fais cela 20 fois!

Avec FOR...NEXT, le compteur est créé, incrémenté et testé implicitement.

Si tu veux seulement le carré des nombres impairs, il faudra ajouter STEP 2 (le pas) à l'instruction no 20:

```
20 FOR I=1 TO 19 STEP 2
```

Fais l'expérience... mets ta boucle "au pas"!

QUESTION Les instructions FOR... NEXT remplacent avantageusement le compteur. Cette notion de compteur est donc inutile!

Réponse: Cet exemple a été choisi justement pour que tu constates qu'une boucle d'itération avec FOR...NEXT est préférable à un compteur...(économie de mémoires). L'important est de comprendre comment fonctionne un compteur dans un programme car tu auras à t'en servir souvent, soit à l'intérieur d'une boucle FOR...NEXT ou à l'extérieur...

Les couleurs du CADRE et du FOND de l'écran

Dans son état normal, le C64 donne un fond d'écran de couleur bleu foncé (No 6 p.74) et un cadre bleu clair (No 15 p.74).

Le REGISTRE 53280 contient l'une des 16 couleurs du **cadre** de l'écran

Le REGISTRE 53281 contient l'une des 16 couleurs du **fond** de l'écran

VOIR le tableau des codes-couleurs d'écran page 38. Le numéro de ces couleurs est introduit dans les mémoires 53280 et 53281 par l'instruction POKE.

EXERCICE 8-A Mets un **cadre** orange et un **fond** noir:
POKE 53280,8:POKE 53281,0   RETURN

EXERCICE 8-B Demande au C64 d'imprimer les **caractères** en blanc:

PRINT"■"   ou [PRINT CHR$(5) (Voir No 5 p.76)].
(↖ NO 2 PAGE 74)

Cette combinaison te permet d'écrire en blanc sur fond noir avec cadre orange. Essaie d'autres combinaisons...

L'instruction précédente était en mode direct. Tu peux aussi mettre cette instruction dans un programme. Lis le programme C-2 (page 85) **et note les instructions** dans lesquelles le cadre ou le fond de l'écran sont transformés
En as-tu trouvé au moins 4?

EXERCICE 9-A Programme qui fait passer le Fond de l'Ecran par ses 16 couleurs...avec une seconde d'attente...

```
10 FOR FE=0 TO 15
20 POKE 53281,FE
30 FOR T=1 TO 1000:NEXT T
40 NEXT FE
```

EXERCICE 9-B Programme qui fait passer le CAdre par ses 16 couleurs → avec une seconde d'attente entre chaque couleur:

```
10 FOR CA=0 TO 15
20 POKE 53280,CA
30 FOR T=1 TO 1000:NEXT T
40 NEXT CA
```

EXERCICE 9-C Construis toi-même un programme qui donne les 16 couleurs **des caractères** (Voir exercice 5-B).

LET

L'instruction LET est une instruction **d'affectation.** Elle est toujours disponible sur le C64 mais tu n'as pas à l'utiliser dans tes programmes car cette instruction est si fréquente que les constructeurs ont organisé le "système" du C64 pour que LET joue son rôle automatiquement. En principe, il faudrait écrire ainsi ce petit programme:

```
10 LET A=3                          L'écran affichera:
20 LET B$="MOIS"                       3          mois
30 PRINT A,B$
```

LET indique que tu veux mettre la valeur 3 dans la variable numérique A (affecter 3 à A) et mettre la chaîne "MOIS" dans la variable alphanumérique (ou variable chaîne) B$.

EXERCICE 10 Ecris le petit programme ci-dessus sans employer l'instruction LET

```
10 A=3
20 B$="MOIS"
30 PRINT A,B$                    RUN et RETURN
```

Il est important de noter que, dans le langage BASIC, le **signe = est un signe d'affectation.** Il n'a pas la signification d'une équation algébrique.

Un autre exemple: Regarde l'exercice 6, ligne 60, page 80: Tu as I=I+1. Il faut lire cette instruction de la façon suivante: **Ajoute 1 à la variable I et met (affecte) le résultat dans la variable de gauche qui se trouve à être aussi la variable I.** Si la variable I valait 4 avant l'affectation, eh bien quand l'instruction sera exécutée, l'ancien contenu sera perdu et I prendra la valeur 5 (note: ceci nous **permet d'utiliser des formules de récurrence** pour calculer, par exemple, la somme des N premiers nombres).

Le langage BASIC est une adaptation du langage FORTRAN pour rendre plus faciles les manipulations de caractères. Dans nos 2 exemples ci-dessus, A=3 et I=I+1 le FORTRAN écrit A ← 3 et I ← I+1. Dans ce cas, la flèche traduit mieux la réalité de l'affectation que le signe = .
Note: Basic est un langage développé par le Darmouth College en 1960 et les 5 lettres de mot **Basic** viennent de: Beginner's all purpose symbolic instruction code.

PROJET C-2

Tu as déjà sur ta cassette la PROJET C-1 de la page 30.

a) Tu peux introduire le dessin des oiseaux dans la mémoire du C64 et insérer (INST) une variable chaîne devant chacun des guillemets des lignes 100 à 240.

b) Mets le fond de l'écran et le cadre de couleur bleu clair et les oiseaux de couleur noire.

c) Fais avancer l'ensemble des oiseaux vers la droite de l'écran au rythme d'un battement d'ailes par colonne.

Pour cette animation utilise l'instruction PRINT avec les caractères de COMMANDE d'AFFICHAGE du curseur et du CLR (nos 1, 3, 5, 6 page **35**).

Utilise le caractère "curseur à droite" comme compteur de colonnes. Dans ce cas la variable du compteur doit être une variable chaîne de caractères.

d) Fais scintiller un soleil en haut de l'écran sur le coin droit en te servant de l'instruction POKE.

e) Des scientifiques pensent que les oiseaux migrateurs s'orientent le jour d'après le soleil... et la nuit d'après les étoiles...

Quand les oiseaux atteignent le milieu de l'écran:
--mets les oiseaux de couleurs bleu foncé
--mets en noir le cadre et le fond de l'écran
--fais scintiller une étoile à la place du soleil.

f) Quand l'oiseau qui forme la pointe du V a atteint la colonne 38, tu arrêtes le mouvement pendant une seconde (n'oublie pas qu'il y a 40 colonnes mais que la première commence à 0 et la dernière est à 39).

g) Ajoute un compteur (avec variable numérique) pour que le même parcours horizontal de l'écran se répète 2 fois.

Une solution de ce projet est donnée à la page suivante.

```
5 J=0                                 PROGRAMME C-2
10 I$="": E=1024: C=55296
15 POKE 53280,14: POKE 53281,14
20 PRINT"▯▮▮"
25 POKE E+36,77:POKE E+37,66:POKE E+38,78
30 POKE E+75,67:POKE E+77,81:POKE E+79,67
35 POKE E+116,78:POKE E+117,93:POKE E+118,77
40 POKE E+157,93
45 IF I$>="▮▮▮▮▮▮▮▮▮▮▮▮" THEN POKE 53280,0: POKE 53281,0
50 POKE C+36,8: POKE C+37,8: POKE C+38,8
55 POKE C+75,8: POKE C+77,2: POKE C+79,8
60 POKE C+116,8: POKE C+117,8: POKE C+118,8
65 POKE C+157,8
70 POKE C+36,1: POKE C+37,1: POKE C+38,1
75 POKE C+75,1: POKE C+77,1: POKE C+79,1
80 POKE C+116,1: POKE C+117,1: POKE C+118,1
85 POKE C+157,1
90 IF I$<="▮▮▮▮▮▮▮▮▮▮▮" THEN PRINT"▮": GOTO 100
95 PRINT"▮"
100 PRINTI$" ╲
110 PRINTI$" ●
120 PRINTI$" ╱   ╲
130 PRINTI$"     ●
140 PRINTI$"     ╱   ╲
150 PRINTI$"         ●
160 PRINTI$"         ╱   ╲
170 PRINTI$"             ●
180 PRINTI$"         ╲   ╱
190 PRINTI$"         ●
200 PRINTI$"     ╲   ╱
210 PRINTI$"     ●
220 PRINTI$" ╲   ╱
230 PRINTI$" ●
240 PRINTI$" ╱
250 FOR T=1 TO 20: NEXT
255 PRINT"▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮"
270 IF I$="▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮" THEN 460
280 PRINTI$" ╱
290 PRINTI$" ●
300 PRINTI$" ╲   ╱
310 PRINTI$"     ●
320 PRINTI$"     ╲   ╱
330 PRINTI$"         ●
340 PRINTI$"         ╲   ╱
350 PRINTI$"             ●
360 PRINTI$"         ╱   ╲
370 PRINTI$"         ●
380 PRINTI$"     ╱   ╲
390 PRINTI$"     ●
400 PRINTI$" ╱   ╲
410 PRINTI$" ●
420 PRINTI$" ╲
440 I$=I$+"▮▮"
450 GOTO 20
460 J=J+1
470 FOR T=1 TO 1000: NEXT
480 IF J=2 THEN 510
490 POKE 53280,14: POKE 53281,14
500 GOTO 10
510 GOTO 510
```

Le rôle de chacune des lignes de ce programe C-2 est expliqué à la p.86

Rôle de chacune des lignes du programme C-2, page 85

Ligne 5: Initialisation à 0 du compteur numérique. Le compteur augmentera de 1 à la ligne 460.

Ligne 10: Initialisation avec une chaîne vide (des guillemets qui se suivent) du compteur chaîne de caractères. Le compteur augmente de 1 caractère à la ligne 440. Ce caractère fait avancer le curseur d'une colonne (no 3 page 35).

NOTE: Les lignes 5 et 10 peuvent être enlevées du programme parce que ces variables de contrôle sont initialisées à 0 et avec une chaîne vide. Dans ce cas le C64 se charge de les initialiser. Je te conseille de laisser quand même ces lignes pour faciliter la compréhension du listing.

Ligne 15: Les mémoires 53280 et 53281 sont les registres pour la couleur du fond de l'écran et du cadre. Avec POKE tu introduis les valeurs 14 et 14(bleu clair) dans ces mémoires

Ligne 20: Efface l'écran et descendsle curseur de 2 lignes (Nos 1 et 5 p. 35)

Lignes 25à40: Elles affichent un soleil et ses rayons au coin droit de l'écran (voir les codes-caractères d'écran page 75).

Ligne 45: Test (IF...) pour savoir si le curseur (ici l'ensemble des oiseaux) a avancé de 11 colonnes ou plus (>=). Si OUI alors mettre 0 dans les registres 53280 et 53281 (nuit) et aller à la ligne 70. Si NON continuer le programme à la ligne suivante.

Lignes 50 et 65: Elles mettent le soleil de couleur rouge et les rayons oranges (voir tab. des codes-couleurs d'écran page 38).

Lignes 70 à 85: Le soleil devient une étoile (nuit) avec la couleur blanche.

Ligne 90: Test (IF...) pour savoir si le curseur (ici l'ensemble des oiseaux) a avancé de 10 colonnes ou moins (<=). Si OUI alors imprimer les oiseaux de couleur noire et aller à la ligne 100. Si NON le programme continue à la ligne 95: (c'est la nuit).

Ligne 95: Imprime les oiseaux de couleur bleu foncé (voir le no 6 dans le tableau des caractères de commande des couleurs page 74).

Lignes 100 à 240: Forment un bloc qui imprime les oiseaux avec les ailes tournées vers l'arrière. La variable I$ fait avancer le curseur à droite une fois à chaque itération. Elle doit donc apparaître après chaque PRINT pour que les 7 oiseaux se déplacent ensemble vers la droite.

Ligne 250: Boucle d'attente de 20/1000 seconde.

Ligne 255: Remonte le curseur de 16 lignes.

Ligne 270: Si la variable I$ du compteur de colonnes vaut '25 fois curseur à droite' alors aller à 460 car l'oiseau qui forme la pointe du V a atteint l'avant dernière colonne.

Lignes 280 à 420: Forment un bloc qui affiche les oiseaux avec les ailes tournées vers l'avant. L'affichage des oiseaux se trouvera à coincider avec le dessin des lignes 100 à 240 à cause du curseur qui a été remonté (voir ligne 255). Nous aurons ainsi un battement d'ailes par colonne.

Ligne 430: Comme à la ligne 250.

Ligne 440: La variable I$ augmente de 1 caractère (voir "notion de compteur" page 80).

Ligne 450: Reprend le programme à la ligne 20 (nouvelle itération) jusqu'à ce que le TEST de la ligne 270 soit VRAI.

Ligne 460: La variable J augmente de 1.

Ligne 470: Boucle d'attente de 1 seconde.

Ligne 480: Même chose qu'à la ligne 15.

Ligne 490: Si les oiseaux ont parcouru 2 fois l'écran alors......ligne 510.

Ligne 500: **Sinon** recommencer le mouvement depuis le début avec 7 oiseaux à gauche de l'écran.

Ligne 510: Boucle infinie qui empêche l'affichage de READY.

Comparaison des VITESSES d'EXECUTION entre PRINT et POKE

Dans le programme C-2, page 85, l'animation des oiseaux se faisait avec l'instruction PRINT et les caractères de commande d'affichage (page 35) et de couleurs (page 74). Il a fallu mettre deux boucles d'attente pour ralentir le mouvement...

Dans le programme C-3 de la page suivante, l'animation des oiseaux se fait avec l'instruction POKE et tu verras que le mouvement est plus lent malgré l'élimination des boucles d'attente et l'emploi d'une technique: la balançoire (p. 89) qui permet de diminuer le nombre d'instructions POKE.

On peut conclure que l'animation est plus rapide avec PRINT qu'avec POKE. Cependant il faut remarquer que l'instruction POKE offre plus de souplesse et d'élégance. De plus POKE et PEEK (voir p.99) permettent de faire réagir des objets entre

eux sur l'écran, style arcade, alors que ces réactions seraient très difficiles à obtenir avec PRINT...

```
5 PRINT"□": Z=0
10 FOR X=16 TO 54
20 E=1464: N=81
25 IF Z=0 THEN Z=1: L1=73: L2=75: GOTO35
30 Z=0: L1=85: L2=74
35 POKE E+X,N: POKE E+X-40,L1:POKEE+X+40,L2
40 POKE E+X-84,N: POKE E+X-124,L1: POKE E+X-44,L2
45 POKE E+X+76,N: POKE E+X+36,L1: POKE E+X+116,L2
50 POKE E+X-168,N: POKE E+X-208,L1: POKE E+X-128,L2
55 POKE E+X+152,N: POKE E+X+112,L1: POKE E+X+192,L2
60 POKE E+X-252,N: POKE E+X-292,L1: POKE E+X-212,L2
65 POKE E+X+228,N: POKE E+X+188,L1: POKE E+X+268,L2
70 IF N=32 THEN 130
80 IF E=55736 THEN 120
100 E=55736: N=1: L1=1: L2=1
110 GOTO 35
120 E=1464: N=32: L1=32: L2=32
125 IF X=37 AND W=1 THEN END
127 GOTO 35
130 NEXT X
135 W=W+1
140 GOTO 10
```

EXERCICE 11-A Remplis l'écran avec des caractères coeur (rouge) en utilisant l'instruction PRINT et fais imprimer (en blanc) le temps requis pour remplir l'écran

```
5 PRINT"□";
10 Y=TI/60                                  (voir: horloge p.66)
15 FOR I=0 TO 999: PRINT"■♥";: NEXT
20 PRINT"□■"TI/60-Y"SECONDES"  ——————>    2,87 secondes
```

EXERCICE 11-B Remplace PRINT par POKE dans l'ex. 11-A, ligne 15

```
5 PRINT"□";
10 Y=TI/60
15 FOR I=0 TO 999: POKE 1024+I,83: POKE 55296+I,2: NEXT
20 PRINT"□"TI/60-Y"SECONDES"  ——————>    19,50 secondes
```

Une technique: LA BALANCOIRE

La technique de la balançoire permet d'obtenir **alternativement** OUI et NON lors du passage par l'instruction de TEST (IF...). Cette alternance est rendue possible grace à une **variable auxiliaire** qui prendra la valeur un après OUI et zéro après NON.

Nous utiliserons cette technique dans le programme C-3 (animation des oiseaux avec POKE page 88). Les ailes seront alternativement vers l'arrière puis vers l'avant selon que la variable auxiliaire Z sera égale à zéro ou à un.

L'organigramme ci-dessous t'aidera à mieux comprendre cette technique. "Je veux que les ailes soient tournées vers l'arrière" quand X vaut 16, 18, 20, 22,... 54 "et tournées vers l'avant" quand X vaut 17, 19 , 21 , 23 ... 53 .

Explications page 90.
Lignes du programme C-3

Organigramme de la page 89: (explication)

Ligne 5: La variable auxiliaire Z prend, au départ, la valeur 0

Ligne 10: La variable X est initialisée à 16; cette colonne 16 est celle de l'oiseau qui se trouve à la pointe du V.

Ligne 20: E contient la mémoire d'écran de la colonne 0 rangée 10 (il y a une rangée 0!) et N contient le code-caractère 81 qui forme le corps de l'oiseau.

Ligne 25: Instruction de TEST (IF): Z est-il égal à 0? A la 1ère itération: X=16 et Z=0 donc le test fait "balancer" vers la gauche (OUI) et Z prend la valeur 1. A la 2ième itération: X augmente de 1 et comme Z n'est plus égal à 0, le test fera "balancer" vers la droite (NON) et de nouveau Z prend la valeur 0.

Ligne 30: Si la réponse au test est OUI: Mettre 1 dans Z, 73 dans L1 et 75 dans L2. Les caractères 73 et 75 forment les ailes des oiseaux (ailes tournées vers l'arrière). Si la réponse est NON: mettre 0 dans Z, 85 dans L1 et 74 dans L2. Les caractères 85 et 74 forment les ailes tournées vers l'avant. Voir page 75.

Conclusion: Chaque fois que la variable X contiendra un nombre pair, les ailes seront tournées vers l'arrière et avec un nombre impair les ailes seront tournées vers l'avant.

Il faut remarquer que la même série d'instructions POKE (lignes 35 à 65) est parcourue 3 fois à l'intérieur de la boucle: (gain de mémoires)

1- affichage des oiseaux avec les ailes tournées vers l'arrière
2- les oiseaux reçoivent la couleur rouge (code 2 page 38).
3- les oiseaux sont effacés (caractère 32).

EXERCICE 12 Utilise la technique de la "balançoire" pour que les 2 caractères graphiques (SHIFT P inversé et rouge) et (SHIFT M) s'imprimemt alternativement à la suite de la séquence précédente, sur 20 rangées et avec retour chariot à chaque 'PRINT' (2 fois le même caractère à chaque itération)

```
10 X=0
20 FOR I=1 TO 20
30 IF X=0 THEN X=1: D$=D$+"■■TT": GOTO 60
40 X=0
50 D$=D$+"\\"
60 PRINT D$
70 NEXT I: PRINT"□"
```

(No 15; No 3 p. 74)

Tu reconnais la technique de la balançoire qui permet alternativement d'imprimer ⌝ ⌝ et ⧅ ⧅

OUI — premier caractère; NON — deuxième caractère; OUI — premier caractère; NON — deuxième caractère

Les OPERATEURS LOGIQUES

Les explications seront assez brèves dans ce premier volume. Il y a 3 opérateurs logiques qu'on appelle aussi opérateurs booléens: NOT, AND, OR. Ces opérateurs prennent place dans une instruction IF...

1- L'opérateur négation logique NOT (non). Exemple:
IF NOT A THEN 60 Branchement à la ligne 60 si le contenu de la variable A est faux.

2- L'opérateur logique AND (et). Exemple:
IF B=21 AND C=28 THEN 40 Si les valeurs de B et C sont vraies simultanément, cette expression logique sera vraie et il y aura branchemet à la ligne 40. (Voir chapitre V pour avoir de plus amples informations).

3- L'opérateur logique OR (ou). Exemple:
Remplace la ligne 25 du programme C-3 (page 88) par:

```
25 IF X=8 OR X=10 OR X=12 OR X=14 OR X=16 OR X=18 OR X=20 OR
X=22 OR X=24 OR X=26 OR X=28 THEN L1=73:L2=75:GOTO 35
```

NOTE Tu constates que ta ligne 25 contient plus de 80 caractères. Une ligne de 80 caractères correspond à 2 lignes sur l'écran du C64 (2 X 40); le No de ligne est inclus dans ces 80 caractères. Quoi faire avec cette ligne 25?

Il y a 2 solutions:

a) place le 'IF' immédiatement après 25 et ne laisse aucun espace dans ta ligne...Si celle-ci possède encore plus de 80 caractères, alors écris les instructions en abrégé: (G et SHIFT O) pour GOTO (T et SHIFT H) pour THEN...

b) 2ième solution: si le nombre de caractères est toujours supérieur à 80, alors partage ta ligne 25 avec une ligne 26 qui commencerait aussi par IF avec les mêmes valeurs réécrites après THEN dans les deux lignes.

Lignes 25 et 26: Si une des onze valeurs de la variable logique X est vraie ALORS L1=73:L2=75:ALLER à 35
Si aucune n'est vraie l'exécution du programme continue à la ligne 30.
30 L1=85:L2=74 (ailes vers l'avant).

Ce temps consacré aux lignes 25 et 26 te sera utile. Cependant **je te conseille, dans un cas semblable, d'utiliser la technique de la balançoire.** (Voir chap.5 pour d'autres explications sur le OU logique)

Etude de six fonctions: FRE, SPC, TAB, ASC, CHR$, PEEK

SPC( ) et FRE( )

Une fonction est toujours suivie de parenthèses entre lesquelles tu mets l'argument de cette fonction.

La fonction SPC, espace, (anglais: SPaCe) met des caractères 'espaces' à partir de la **position courante du curseur**. L'argument est le nombre d'espaces que tu veux créer (0 à 255).

EXERCICE 13 Mets le C64 à 'ON' et écris ce programme d'édition:

```
10 PRINT"BULLETIN"SPC(20)"SECONDAIRE V"
20 PRINT"ELEVES"SPC(18)"MOYENNE   R/5"
```

Ecris: **PRINT FRE(0)+65536** RETURN

L'écran affiche 38841. C'est le nombre d'octets (bytes) disponibles.(FREE)

Tu peux obtenir le même format d'édition en utilisant la barre d'espacement pour mettre des espaces. (**Note:** PRINT SPC(20) fait imprimer 20 espaces. Donc il positionne le curseur à 20 + 1).

EXERCICE 14-A Fais l'expérience:

```
10 PRINT"BULLETIN                    SECONDAIRE V"
20 PRINT"ELEVES                  MOYENNE   R/5"
```

Ecris: **PRINT FRE(0)+65536** RETURN

L'écran affiche 38815... nombre d'octets disponibles.

La fonction FRE, libre, (anglais: FREe) te permet de demander au C64 combien il lui reste d'emplacements mémoire disponibles pour ton programme. L'argument (0) peut être remplacé par un nombre quelconque.

Dans le premier cas: 38841 octets libres
Dans le deuxième cas: 38815 octets libres

Donc en utilisant la fonction SPC **tu as sauvé** 26 **emplacements mémoires** dans 1 seule ligne BASIC. Chaque fois que tu appuies la barre d'espaces tu prends une mémoire... Si tu as plusieurs espaces à imprimer il est préférable d'utiliser la fonction SPC.

NOTES 1- SPC(0) n'a aucun effet.
2- Si l'argument est une expression non entière le C64 convertira l'expression en un entier.
3- Pour obtenir plus de 255 espaces, par exemple 400,

→ Voir EXERCICE 14-B page 180

tu écris: SPC(255), SPC(145)

4- **L'expression suivante** te dira à chaque instant le nombre de MEV (RAM): **PRINT FRE(0)-(FRE(0)<0)*65536**

TAB( )

L'argument (paramètre) de la fonction TAB est une variable ou une constante entière (0 à 255) qui indique la position où l'on doit commencer une impression. Si le paramètre est < 0 le C64 indiquera ce message: ILLEGAL QUANTITY

Son rôle est semblable au TABulateur de la machine à écrire.

EXERCICE 15

```
5 PRINT"♫"
10 PRINT"ACHATS"TAB(10)"PRIX"TAB(20)"VENTES"TAB(30)"PRIX"
```

Cette dernière instruction imprimera <u>prix</u> à partir de la colonne 10 puis imprimera <u>vente</u> à partir de la colonne 20...

Il ne peut y avoir superposition de deux impressions. Par exemple si par erreur tu écris TAB(2) au lieu de TAB(20), <u>VENTE</u> s'imprimera immédiatement après prix. Lorsque tu utilises <u>plusieurs fois</u> la fonction TAB dans une même instruction, l'argument de chacun des TAB correspond à la colonne 0 du début de la ligne.

<u>Note</u> Lorsque le paramètre de TAB amène le curseur sur une 2e rangée de l'écran, le paramètre du TAB suivant comptera à partir du début de cette 2e rangée.

L'argument de la fonction TAB peut être une <u>expression arithmétique</u>. Ceci permet de faire des tracés approximatifs de fonctions. **Seule la partie entière sera utilisée** pour définir la position du tabulateur.

EXERCICE 16 Fais imprimer sur l'écran le tracé d'une courbe ayant l'allure d'une parabole.

```
10 PRINT"♫": C=4
15 FOR I=11 TO 1 STEP-1
20 PRINT TAB(I*I/C)"*"
25 NEXT I
30 FOR J=1 TO 11
35 PRINT TAB(J*J/C)"*"
40 NEXT J
```

EXERCICE 17 On utilise la fonction TAB pour déplacer des objets sur l'écran (méthode des PRINT). En voici un exemple dans ce programme qui déplace un tank de gauche à droite et de droite à gauche:

```
10 PRINT"▯▮▮▮▮▬ A GAUCHE"
11 PRINT"▮▮H▬ A DROITE
12 PRINT"▮▮APPUIE UNE TOUCHE
20 GETA$:IFA$=""THEN 20
25 PRINT"▯▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮": POKE650,128
30 IFC<0THENC=0
35 IFC>33THENC=33
40 PRINTTAB(C)"   | "
45 PRINTTAB(C)"  ()"
50 PRINTTAB(C)"/‾‾‾‾‾\"
55 PRINTTAB(C)"L_____J"
60 PRINTTAB(C)"OOOOOO"
65 PRINT"TTTTT";
70 GETB$:IFB$="G" OR B$="H" THEN 75
72 GOTO 70
75 PRINTTAB(C)"     "
80 PRINTTAB(C)"     "
85 PRINTTAB(C)"       "
90 PRINTTAB(C)"       "
95 PRINTTAB(C)"       "
100 PRINT"TTTTT";
110 IF B$="G" THEN C=C-1: GOTO 30
120 IF B$="H" THEN C=C+1: GOTO 30
```

Le hic de ce programme réside dans la fonction TAB! L'argument de TAB(C) est la variable C qui peut prendre les valeurs 0 à 35 et 35 à 0 et ainsi déplacer tout le dessin à droite ou à gauche sur l'écran.

Lorsque le dessin occupe une nouvelle position sur l'écran, il faut que le dessin précédent soit effacé. C'est le rôle des lignes 75 à 95.

Les lignes 110 et 120 changent la valeur de l'argument C en plus ou en moins selon que le tank se dirige vers la droite ou vers la gauche.

La ligne 30 empêche l'argument C d'atteindre une valeur inférieure à zéro, ce qui provoquerait un message d'erreur (illegal quantity). La ligne 35 empêche l'argument C d'atteindre une valeur supérieure à 35, ce qui ferait dérouler l'écran et déformer le dessin.

Il faut noter aussi la mémoire 650 de la ligne 25. Cette adresse 650 contient normalement la valeur 0 mais quand tu mets 128 dans **la mémoire 650**, tu **rends toutes les touches du C64 "répétitives"**...(comme les touches CRSR). Si tu tiens enfoncée l'une ou l'autre des touches G et H, le tank se déplacera jusqu'à la colonne zéro ou jusqu'à la colonne 35 (arrière du tank). **Mets sur cassette** le programme de l'exercice 17.

L'instruction GOSUB (chapitre IV) **te permettra d'économiser** plusieurs mémoires lorsque tu transformeras le programme ci-dessus.

ASC( )

Le code ASCII (American Standard for Communication Information Interchange) est le code américain pour communiquer et échanger des informations

La fonction ASC, comme toutes les autres fonctions BASIC, est suivie de parenthèses entre lesquelles tu mets un argument. L'argument de ASC est l'un des caractères alphanumériques du tableau de la page 76. N'oublie pas que **ces caractères doivent être placés entre des guillemets.**

EXERCICE 18 Ecris:

```
PRINT ASC("A")                      RETURN
```

L'écran affichera 65. Consulte le tableau de la page 76 et tu verras que 65 correspond bien à la lettre A.

EXERCICE 19 Ecris en mode direct:

```
A$="ANDRE": PRINT ASC(A$)           RETURN
```

L'écran affichera encore 65 car la fonction ASC fournit une valeur numérique qui est le code ASCII **du premier caractère (octet) de la chaîne**.

Remarque Dans le tableau de la page 76 il faut savoir que 96 à 127 et 192 à 223, sont équivalents. De même 126 est équivalent à 255. Ainsi le code du caractère "pique" est 97 et aussi 193.

EXERCICE 20 Ecris:

```
PRINT ASC("♠")                RETURN.
            (SHIFT A)
```

L'écran affichera la plus grande valeur, soit 193

CHR$( )

L'argument de la fonction CARACTERE CHR$ (anglais: CHaRacter) est l'un des codes (un nombre) du tableau de la page 76.

Avec PRINT tu obtiendras le caractère chaîne correspondant au code de l'argument.

EXERCICE 21 Ecris:

```
PRINT CHR$(97)                RETURN.
```

L'écran affichera le caractère chaîne: **pique.** Tu obtiendras aussi un pique avec CHR$(193).

Je te conseille de faire plusieurs exercices pour te familiariser avec cette fonction. Utilise différentes valeurs du tableau de la page 76 et vérifie les résultats.

Exemple: Il y a 4 touches à droite du clavier: f1, f3, f5, f7
et avec SHIFT: f2, f4, f6, f8

EXERCICE 22 Ecris un petit programme qui attend la touche f1:

```
10 GET A$:IF A$ <> CHR$(133) THEN 10
20 PRINT"TU VIENS D'APPUYER SUR LA TOUCHE F1"
```

Appuie sur différentes touches et tu constateras que seule la touche f1 permet au C64 d'exécuter la ligne 20.

Autre exemple: la seule façon d'imprimer le caractère guillemet dans une chaîne de caractères est d'utiliser la fonction CHR$. Consulte le tableau (page 76) et tu trouveras le code 34 pour les guillemets. Exemple

EXERCICE 23 Ecris:

```
10 PRINT"ELEVES SECONDAIRE III"
20 PRINT"  "CHR$(34)"         "CHR$(34)"  (7 espaces)  IV"
30 PRINT"  "CHR$(34)"         "CHR$(34)"  (8 espaces)   V"
```

Tu peux employer la fonction CHR$ pour imprimer tous les caractères du tableau de la page 35.

Il faut bien noter que si le paramètre de la fonction CHR$( ) est le code d'un des caractères de COMMANDE d'affichage ou de couleur, ces caractères ne seront pas affichés sur l'écran mais ils joueront quand même leur rôle au moment de l'exécution du programme. Exemple:

EXERCICE 24 Ecris:

```
10 PRINT CHR$(147)
20 PRINT CHR$(18)"PIERRE"
30 PRINT CHR$(145)CHR$(5)"JOSEPH"
```

Vérifie dans le tableau page 76: 147 efface l'écran; 18 met en contraste inversé; 145 remonte le curseur d'une rangée; 5 donne la couleur blanche.

EXERCICE 25 Ce programme trace une ligne horizontale qui s'imprime sur les 40 colonnes de la rangée 25 **sans** faire dérouler l'écran.

(23 CRSR bas)

```
5 PRINT"[illegible]"
10 FOR I=1 TO 40: IF I=40 THEN PRINT CHR$(157)CHR$(148);
20 PRINT"‾";
30 NEXT I
40 GOTO 40
```

Le code 157 fait reculer le curseur d'une case; le code 148 est celui de la touche INST; donc CHR$(148) fait une insertion, c'est-à-dire qu'il déplace d'une case à droite le caractère sur lequel le curseur se trouve positionné. Un trait horizontal se trouve maintenant sur la dernière case (1000) et le curseur est sur la case précédente. Lorsque le programme exécute la ligne 20, il met un trait à l'endroit préparé pour l'insertion (case 999) et le curseur se déplace sur la case suivante.

NOTE **L'emploie de INST introduit automatiquement le mode guillemet.**

EXERCICE 26 Change la ligne 10 de l'exercice 25 en utilisant des caractères de COMMANDE au lieu de la fonction CHR$

```
10 FOR I=1 TO 40: IF I=40 THEN PRINT"▮▮▮";
```

Même s'ils se ressemblent, les deux caractères entre les guillemets sont différents. Le premier est le no 4 page 35 et le deuxième est le no 10 page 35. Le symbole de ce caractère no 10 est un rectangle en contraste inversé avec une colonne éteinte (l'avant dernière, à gauche); c'est ce caractère qui permet de faire des insertions à l'intérieur d'un programme.

Avant de passer à l'étude de la fonction PEEK, tu vas faire un exercice plus élaboré qui résumera plusieurs notions apprises...

EXERCICE 27 Fais imprimer 3 cartes de même format et laisse 5 secondes d'attente avant d'effacer le dessin précédent: **1re carte**, soirée dansante; **2e carte**, un deuil; **3e carte**, bonne fête.

```
5 PRINT"▯": POKE 53281,1
10 X$=CHR$(28)+CHR$(42): REM X$="▮*"
20 FOR I=1 TO 3
30 IF I=2 THEN X$=CHR$(18)+CHR$(144)+CHR$(32): REM X$="▮▮ "
40 IF I=3 THEN X$=CHR$(31)+CHR$(166): REM X$="▮▮"
50 PRINT"▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮"; ──→(HOME, 5 CRSR bas, 10 CRSR droite)
60 FOR J=1 TO 20: PRINT X$;: FORT=1TO100: NEXT: NEXT
70 FOR J=1 TO 10: PRINT X$"▮▮▮";: FOR T=1 TO 100: NEXT: NEXT
80 FOR J=1 TO 20: PRINT X$+"▮▮▮";: FORT=1 TO 100: NEXT: NEXT
90 FOR J=1 TO 10: PRINT X$"▮▮";: FORT=1 TO 100: NEXT: NEXT
100 IF I=1 THEN PRINT"▮▮▮▮▮▮▮▮▮▮"SPC(14)"SOIREE"
110 IF I=1 THEN PRINTSPC(54)"DANSANTE "SPC(71)"21 HEURES"
120 IF I=2 THEN PRINT"▮▮▮▮▮▮▮▮▮▮"SPC(14)"▮QU'IL "
130 IF I=2 THEN PRINTSPC(54)"▮REPOSE  "SPC(72)"EN PAIX! "
140 IF I=3 THEN PRINT"▮▮▮▮▮▮▮▮▮▮"SPC(14)"BONNE  ♠"
150 IF I=3 THEN PRINTSPC(54)"FETE MON"SPC(71)" TOUTOU ♠ "
160 FOR T=1 TO 5000: NEXT
170 NEXT I
180 PRINT"▮▮▮
```

Analyse chacune des lignes de ce programme et ensuite

transforme-le selon tes goûts et ton talent. Tu peux facilement faire mieux!...Tu peux écrire les lignes 100 à 150 sur 3 lignes BASIC au lieu de 6. Nous les avons partagées pour qu'elles n'empiètent pas sur la marge de cette page.

Tu peux dire avec fierté que tu connais maintenant trois méthodes pour obtenir le même caractère sur l'écran. Prenons comme exemple le caractère B

1re méthode: PRINT"B" L'écran affiche B
2e méthode: PRINT CHR$(66) L'écran affiche B (page 76)
3e méthode: POKE 1024,2 (+couleur) L'écran affiche B (p. 75)

↑ A l'origine. Tu peux changer 1024 pour une autre adresse de l'écran...

PEEK( )

L'argument de la fonction PEEK est toujours une des 65535 mémoires du C64. Si tu mets PRINT devant PEEK( ), l'écran affichera le **contenu** de cette mémoire.

Le contenu d'une mémoire est toujours un nombre entier compris entre 0 et 255. C'est un octet (anglais: byte).

EXERCICE 28 Ecris en mode direct

PRINT PEEK (53281) RETURN

L'écran affiche 6. C'est le code de la couleur normale du fond de l'écran: bleu foncé.

EXERCICE 29-A Utilise l'instruction POKE pour mettre le code 15 (gris clair) dans le registre 53281.

POKE 53281,15 RETURN

EXERCICE 29-B Utilise la fonction PEEK pour voir si vraiment le code 6 a été remplacé par 15 dans la mémoire 53281:

PRINT PEEK(53281) RETURN. L'écran affiche 15.

Avant de passer au PROJET D-2, nous allons solutionner un problème laissé en suspend au milieu de la page 43. Il serait bon que tu relises l'exercice 9 page 42 où l'affichage du 40e coeur faisait dérouler l'écran.

Voici cette 3e solution avec la fonction PEEK. La mémoire 214 contient le numéro de ligne de la position du curseur. Quand le curseur imprime la ligne 24 il suffit de lui dire qu'il est sur la ligne 23.

Lorsque le C64 imprimera la rangée 24 (la dernière), <u>il ne fera pas dérouler l'écran</u> car il se croira sur la ligne 23! Il pourrait t'arriver quelque chose de semblable si tu habites au 24e étages et qu'un malin s'avise de changer les numéros des boutons de l'ascenseur... Tu pourrais aussi te croire au 23e étage!

EXERCICE 30 Ecris le programme suivant

```
5 PRINT CHR$(147);
10 FOR X=1 TO 25
15 IF PEEK(214)=24 THEN POKE 214,23
20 PRINT CHR$(115)
30 NEXT
40 GOTO 40
```

Tu as réussi à tromper le C64 sans le mettre "hors de lui-même"... Mais il y a d'autres tromperies qu'il "prend mal" et alors il va afficher une série de caractères heureusement incompréhensibles... Tu n'as plus d'autres solutions que d'éteindre et rallumer pour retrouver ton C64 en pleine forme car il aura "tout oublié" ... même le beau programme que tu préparais!

Il faut donc être prudent quand tu emploies POKE car c'est toi qui peux te tromper sur l'adresse ou le contenu de l'adresse... Le C64 ne sera jamais endommagé par ces erreurs. La prudence est recommandée simplement pour éviter la perte du programme que tu es en train de batir.

Exemple: la mémoire 788 contient normalement la valeur 191. Contrôle cette affirmation en écrivant: PRINT PEEK (788) RETURN. Ensuite entre (load) un des programmes que tu as implantés sur ta cassette (évidemment le programme de ta cassette demeurera intact).

Ecris: POKE 780,0 RETURN. Ecris: LIST... RUN... Rien à faire, l'insulte est trop sérieuse! il faut éteindre et rallumer. Il arrive souvent que l'erreur ne soit pas fatale; alors tu peux récupérer ton programme si tu appuies **<u>simultanément</u>** <u>sur STOP et RESTORE</u>.

EXERCICE 31 Ecris un programme qui remplit l'écran avec 1000 petites cases bleues. Utilise la fonction PEEK pour éviter le déroulement de l'écran!

```
5 PRINT"♥";
10 FOR I=1 TO 1000
15 IF PEEK(214)=24 THEN POKE 214,23
20 PRINT "┐";
30 NEXT    ←——(SHIFT P)
40 GOTO 40
```

L'écran est rempli de petits rectangles bleus. Cet exercice permet de visualiser les 1000 cases du bloc mémoires

écran (25 X 40). Efface la ligne 15 et tu verras l'écran remonter de 4 rangées, c'est-à-dire d'une ligne BASIC.

EXERCICE 31-B : Voir page 180

EXERCICE 32-A Le but de cet exercice est expliqué à l'intérieur du programme dans des instructions REM.

```
10 PRINT PEEK(197)
20 H=TI/60
30 IF PEEK(197) <> 64 THEN 30
38 REM ***TEMPS ECOULE PENDANT QUE TU TIENS LA
39 REM TOUCHE 'RETURN'(NO 15) ENFONCEE***
40 PRINT TI/60-H "SECONDE(S)"
50 PRINT PEEK(197)
```

Le tampon clavier prend une importance capitale avec PEEK. Nous y reviendrons dans le chapitre V.

EXERCICE 32-B : La fonction PEEK permet de déceler les collisions...(Voir page 180)

READ DATA RESTORE

Quand tu as une liste de constantes il est souvent avantageux de les écrire dans des instructions DATA et séparer chaque donnée (constante) par une virgule.

Le rôle de l'instruction READ est de lire chacune des données contenues dans les instructions DATA.

Exemple: Supposons que tu as fait cinq devoirs de mathématique et que tu as obtenu les notes suivantes: 80, 75, 78, 69, 85. Supposons aussi que tu prends la variable N pour représenter chacune des notes.

EXERCICE 33 **1er cas:** Ecris seulement les 3 lignes suivantes

```
10 READ N
20 PRINT N
30 END
```

Après RUN et RETURN tu lis sur l'écran: OUT OF DATA
ERROR IN 10

L'ordinateur cherche les données qu'il doit lire (READ) dans une instruction DATA (donnée) mais cette instruction est absente du programme.

Conclusion: Dans un programme, l'instruction READ doit toujours être accompagnée d'une instruction DATA.

EXERCICE 34 **2e cas:** Ajoute une instruction DATA à la ligne 40. Cette instruction contiendra tes cinq notes de mathématique.

```
40 DATA 80, 75, 78, 69, 85
```

L'ordinateur imprime 80 sur l'écran. Tu lui as demandé de lire N, c'est-à-dire la première valeur contenue dans DATA.

NOTE a) L'instruction DATA peut être placée n'importe où dans le programme, même après END ou avant READ.
b) Les instructions DATA sont des instr. **non exécutables.**
c) On ne met pas de virgule après la dernière donnée.

<u>**EXERCICE 35**</u> **3e cas:** Tu veux faire imprimer les 5 données de l'instruction 40. Il faut placer une boucle FOR... NEXT qui comptera jusqu'à cinq.

```
5 FOR I=1 TO 5
10 READ N
20 PRINT N
25 NEXT
30 END
40 DATA 80, 75, 78, 69, 85
```

L'ordinateur lit la première donnée et l'imprime puis à la deuxième itération il lit et imprime la deuxième donnée... et ainsi de suite.

<u>**EXERCICE 36**</u> **4e cas:** Si tu ajoutes une itération à la boucle

```
5 FOR I=1 TO 6
```

Le C64 lit les 5 données contenues dans DATA et les imprime mais quand il arrive à la 6e itération, la 6e donnée étant absente, il imprime: OUT OF DATA
ERROR IN 10

Il manque une ou des données.

Fais l'expérience... Ensuite tu ajouteras une 6e note (te gêne pas, vas-y avec un 100) soit <u>dans un nouveau DATA</u> à la ligne 50 soit <u>à la suite des données de la ligne 40.</u>

Qu'arrivera-t-il si tu ajoutes cette 6e note dans un DATA placé avant la ligne 40, par exemple à la ligne 35? **Fais l'expérience...**

5e cas: Tu veux afficher une appréciation après chacune de tes notes: Excellent pour 90. Très bien pour 80. Bien pour 70. Passable pour 60. Faible pour < 60.

Tu utilises une variable chaîne de caractères, supposons A$, pour représenter les appréciations. Il faut noter que l'instruction DATA mémorise aussi bien des constantes numériques que des constantes chaînes.

Aux lignes 10 et 20 tu ajoutes la variable A$ sans oublier la virgule pour séparer N de A$.

A la ligne 20 tu peux mettre un point virgule entre N et A$ pour imprimer les deux valeurs l'une près de l'autre.

A la ligne 40 ajoute l'appréciation après chacune des notes et tu obtiendras le programme suivant (nous avons partagé les données sur 2 lignes).

EXERCICE 37 Ecris:

```
5 FOR I=1 TO 6
10 READ N,A$
20 PRINT:PRINT N;A$
25 NEXT
30 END
40 DATA 80, TRES BIEN, 75, BIEN, 78 , BIEN, 69, PASSABLE
50 DATA 85, TRES BIEN, 100 EXCELLENT
```

Pour aérer les résultats affichés sur l'écran, il y a une instruction PRINTà la ligne 20 (rien imprimer équivaut à laisser une ligne libre).

Dans ce programme, le C64 lit **deux valeurs** dans DATA **à chaque itération:** la première donnée est un nombre représenté par la variable numérique N et la deuxième est une chaîne de caractères représentée par la variable A$.

REMARQUES 1- Le programme ci-dessus est un exercice (READ... DATA); il y a des moyens plus rapides, plus rationnels et plus économiques (en mémoires) pour indiquer les appréciations...

2- La chaîne de caractères n'a pas besoin de guillemets dans l'instruction DATA (le C64 s'attend à une chaîne) **à moins qu'elle contienne un caractère graphique** ou une virgule ou deux points.

3- La plupart des programmeurs mettent les instructions DATA à la fin du programme, après END.

EXERCICE 38 **6e cas:** Tu veux que ton programme relise les données (dans DATA) une 2e ou une 3e fois...

Si tu recommences la boucle des nos 5 à 25 en mettant GOTO 5 dans 30, qu'obtiendras-tu sur l'écran? **Fais l'expérience...**

Ton C64 affiche les notes et les appréciations mais il refuse de recommencer. Il écrit: OUT OF DATA
ERROR IN 10

La solution est dans **l'instruction RESTORE.**

A chaque lecture d'une donnée dans DATA, l'interpréteur du C64 déplace un pointeur (ou indicateur de lecture) vers la prochaine donnée... et quand la lecture est terminée, le pointeur demeure à la fin des données.

L'instruction **RESTORE** **(rétablir)** remet le pointeur au début de la 1ère instruction DATA du programme et le C64 est prêt à relire les lignes DATA (voir le programme D-2: Québec chante).

EXERCICE 39 Remplace la ligne 30 par celle-ci:

30 RESTORE: GOTO 5   et prépare-toi à appuyer sur STOP

EXERCICE 40 L'ordinateur relit et affiche les données sans arrêt. Si tu veux qu'il lise 2 fois il faut ajouter un compteur et une instruction de TEST entre les lignes 25 et 30 pour sortir de la boucle.

```
3 C=0
5 FOR I=1 TO 6
10 READ N,A$
20 PRINT N;A$
25 NEXT
27 C=C+1: IF C=2 THEN END
28 PRINT
30 RESTORE: GOTO 5
40 DATA 80,TRES BIEN,75,BIEN,78,BIEN,69,PASSABLE
50 DATA 85,TRES BIEN,100,EXCELLENT
```



Photo du projet D-1, pages 30 et 33.

Ce sera plus joli en haute résolution (volume 2)

PROGRAMMATION du SON et de la MUSIQUE

Le processeur 6581 est le coeur du synthétiseur de son/musique (3 voix) de ton COMMODORE 64. Les Américains l'appellent le SID: Sound Interface Device. Le SID possède 29 registres ou mémoires (54272 à 54300) qui contrôlent les trois générateurs de SON. Voir le tableau page 122 .

Pour que ton C64 produise des sons, il faut que tu fasses les **6 opérations** suivantes:

	ADRESSES (VOIX 1)
1- Mettre "du **volume**" (intensité du son)	54296
2- Préciser l'**enveloppe** de l'onde c'est-à-dire les variations d'amplitude...(AD et SR)	54277 et 54278
3- Donner sa **fréquence** (Hte F. et Basse F.)	54273 et 54272
4- Mettre à "**ON**" le registre de la **forme de l'onde.** (Bit zéro à 1)	54276
5- La **durée** du son (**Boucle d'attente...**	
6- Mettre à "**OFF**" le registre de la **forme de l'onde.** (Bit zéro à 0)	54276

EXERCICE 41-A Ecris ce programme et tu entendras un extrait de la chanson: "mon beau sapin" jouée **à l'orgue:**

```
5 REM  ORGUE                               Opérations: (Voir ci-dessus)
10 POKE 54296,15                  ------------------>Opération No 1
20 POKE 54277,0 : POKE 54278,240        ------------>Opération No 2
30 READ HF,BF,DU: IF HF=-1 THEN END
40 POKE 54273,HF: POKE 54272,BF         ------------>Opération No 3
50 POKE 54276,17                  ------------------>Opération No 4
60 FOR T=1 TO DU: NEXT            ------------------>Opération No 5
70 POKE 54276,16: FOR T=1 TO 20:NEXT--------------->Opération N  6
80 GOTO 30
500 DATA 21,31,512,25,30,384,25,30,128,25,30,512
510 DATA 28,49,512,31,165,384,31,165,128,31,165,768
520 DATA 25,30,256,28,49,256,31,165,256,33,135,512
530 DATA 23,181,512,28,49,512,25,30,512
540 DATA -1,-1,-1
```

La ligne 30 lit (READ) 3 données dans DATA: haute fréquence(HF), basse fréquence(BF) et la durée(DU) de la note.

La ligne 80 retourne à la ligne 30 pour lire (READ) les 3 données suivantes...
SI (IF) la première des 3 données (HF) est égale à -1, ALORS (THEN) le programme est terminé (END).

EXERCICE 41-B Le premier registre du SON est à l'adresse 54272. Mets la valeur 54272 dans la variable S (SON) et refais l'exercice 41-A en tenant compte de cette variable. Ca va aider **ta mémoire**!

```
5 S=54272 :   REM  ORGUE
10 POKE S+24,15
20 POKE S+5,0 : POKE S+6,240
30 READ HF,BF,DU: IF HF=-1 THEN END
40 POKE S+1,HF: POKE S,BF
50 POKE S+4,17
60 FOR T=1 TO DU: NEXT
70 POKE S+4,16: FOR T=1 TO 20:NEXT
80 GOTO 30
500 DATA 21,31,512,25,30,384,25,30,128,25,30,512
510 DATA 28,49,512,31,165,384,31,165,128,31,165,768
520 DATA 25,30,256,28,49,256,31,165,256,33,135,512
530 DATA 23,181,512,28,49,512,25,30,512
540 DATA -1,-1,-1
```

EXERCICE 42 Si tu veux que la chanson de l'exercice 41-B soit jouée **au** xylophone, change l'ENVELOPPE de l'ONDE à la ligne 20 en mettant 9 à la place de 0 et de 240:

```
20 POKE S+5,9: POKE S+6,9
```

EXERCICE 43 Pour changer le timbre d'un son, tu changes la forme de l'onde (lignes 50 et 70). On utilise l'onde à dents de scie pour **le violon** parce qu'elle contient toutes les harmoniques. Change les lignes 20, 50 et 70 de l'exercice 41-B:

```
20 POKE S+5,168: POKE S+6,169
50 POKE S+4,33
70 POKE S+4,32: FOR T=1 TO 40:NEXT
```

EXERCICE 44 Utilise l'exercice 43 (Violon avec ondes à dents de scie) et mets 144 et 243 (AD et SR) à la ligne 20. Tu obtiendras un son unique au synthétiseur:

```
20 POKE S+5,144: POKE S+6,243
```

Tu peux inventer tes propres effets sonores

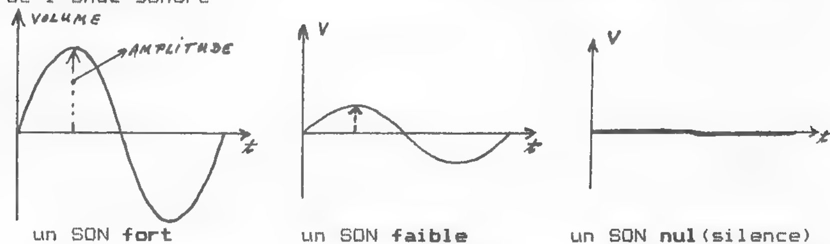
Ton C64 possède un véritable synthétiseur de musique. Que tu sois musicien ou non, tu dois connaître les étonnantes possibilités de ce synthétiseur et te servir de cette merveille dans tes programmes...Ce serait dommage de laisser dormir ce "petit Korg" à l'intérieur de ton C64!

Pour inventer tes propres effets sonores, **étudie les 3 qualités du SON:** A- **intensité** B- **hauteur** C- **timbre.** Tourne la page et **ATTAQUE**!

Les TROIS QUALITES du SON

A- L'intensité d'un SON :

L'intensité d'un son est reliée à **l'amplitude (ou volume)** de l'onde sonore



un SON **fort** un SON **faible** un SON **nul** (silence)

L'intensité du SON **correspond au volume** du SON. Quand tu diminues le volume de la TV pour ne pas déranger ceux qui dorment, tu diminues l'intensité ou l'amplitude **de l'onde.**

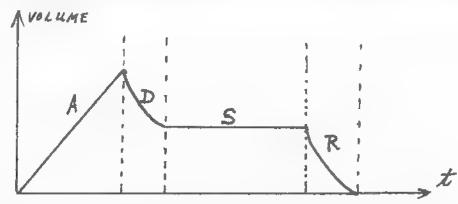
Ton C64 peut **contrôler l'intensité d'un SON** par le **volume principal** et par **l'enveloppe de l'onde:**

1- Le volume principal est à l'adresse 54296 (pour les 3 voix) et tu peux mettre à cette adresse une valeur comprise entre 0 (silence) et 15 (maximum).

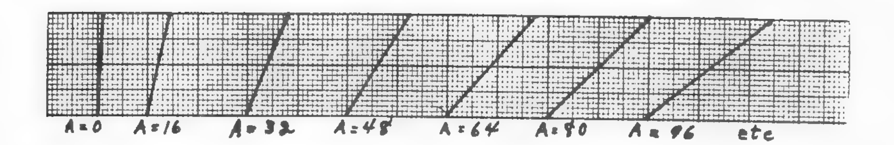
Note importante: tu mets 15 dans 54296 et tu utilises la 2ième manière, beaucoup plus efficace et **raffinée** pour contrôler le volume de l'onde en précisant le profil de son <u>enveloppe</u>:

2- L'enveloppe d'un SON

Le volume d'un SON musical varie depuis le début où il est émis jusqu'à ce qu'il s'évanouisse. Pour mieux préciser l'évolution de l'amplitude on divise l'onde en 4 phases: Attaque (A), Déclin (D), Soutien (S) et Relaxation (R).



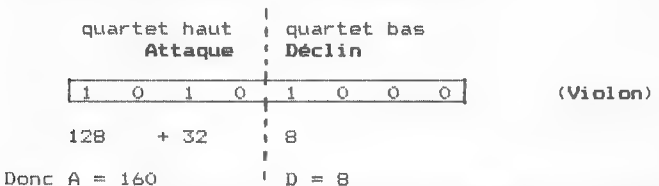
a) L'Attaque (A) est la phase pendant laquelle le volume de la note passe de zéro à sa valeur maximale ou sommet (anglais:peak). **A** peut valoir de 0 à 15 <u>fois 16</u> (0,16,32,48,64,...240).



Exemple: si A = 0 le volume atteint instantanément sa valeur maximale (ex. l'orgue) et si A = 160 (10 X 16) il faut 1/2 seconde pour atteindre le volume maximal (ex. violon).

b) Le Déclin (D) est la phase pendant laquelle le volume de la note tombe à un niveau intermédiaire (anglais:Decay). **D** peut prendre une valeur comprise entre 0 et 15.
Exemple: si D = 0 le volume prend instantanément le niveau de Soutien de la note (ex. trompette) et si D = 9 il faudra environ 3/4 de seconde pour atteindre le niveau de Soutien (ex. piano).

La somme des deux paramètres A + D doit être placée à l'adresse 54277 (voix 1)
Note: L'octet (ou byte) qui contient l'information A+D est formé de 2 quartets (anglais:nybbles).



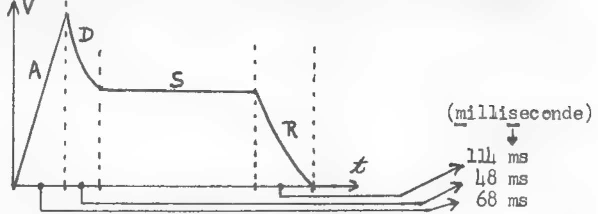
Et **A + D = 168 ⟶ Valeur que tu mets** (POKE) **dans le** registre 54277 (Voix 1)

c) Le Soutien (S) est une valeur qui définit à <u>quelle proportion</u> du volume maximum la note sera <u>maintenue</u> (anglais:Sustain). S peut prendre les valeurs 0 à 15 <u>fois 16</u> (0,16,32,48,64,...240)
Exemple: Si S = 0 il n'y a aucun Soutien (ex. piano, xylophone). Si S = 240 (15 X 16) le Soutien est maintenu au volume maximum (ex. orgue).

d) La Relaxation (R) <u>exprime la rapidité</u> avec laquelle la note atteindra le <u>volume zéro</u>. (anglais:Release). R peut prendre les valeurs 0 à 15.
Exemple: Si R = 0 le volume chute instantanément à zéro (ex. flute, orgue...). Si R = 9 il faudra 3/4 de seconde pour que le volume tombe à zéro (ex. violon)

La somme des deux paramètres **S + R** doit être placée à l'adresse 54278 (Voix 1).
Note: L'octet qui contient **l'information S + R est formé de 2 quartets:** un quartet haut pour S et un quartet bas pour R...**(Voir explications pour A et D ci-dessus)**

Voiciquelle serait l'enveloppe du SON fourni par une **trompette:**

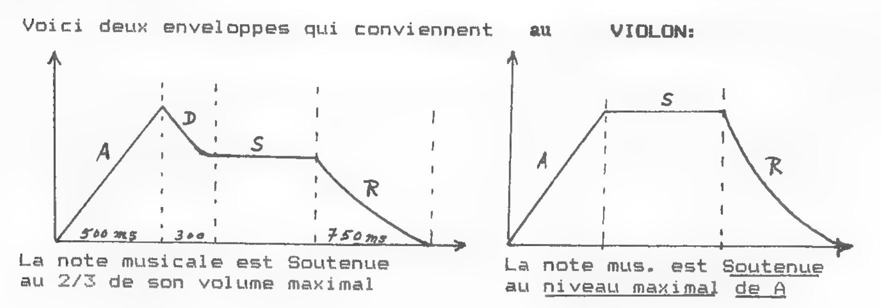


Les valeurs de A, D et R dépendent du temps:

Attaque (A)		Déclin (D) et Relache (R)	
Valeurs	Temps	Valeurs	Temps
0 X 16 = 0	2 ms	0	6 ms
1 X 16 = 16	8 ms	1	24 ms
2 X 16 = 32	16 ms	2	48 ms
3 X 16 = 48	24 ms	3	72 ms
4 X 16 = 64	38 ms	4	114 ms
5 X 16 = 80	56 ms	5	168 ms
6 X 16 = 96	68 ms	6	204 ms
7 X 16 =112	80 ms	7	240 ms
8 X 16 =128	100 ms	8	300 ms
9 X 16 =144	250 ms	9	750 ms
10 X 16 =160	500 ms	10	1.5 s
11 X 16 =176	800 ms	11	2.4 s
12 X 16 =192	1 s	12	3 s
13 X 16 =208	3 s	13	9 s
14 X 16 =224	5 s	14	15 s
15 X 16 =240	8 s	15	24 s

La valeur du Soutien (S) ne dépend pas du temps. S indique à quelle proportion du volume maximal (sommet) la note sera SOUTENUE.

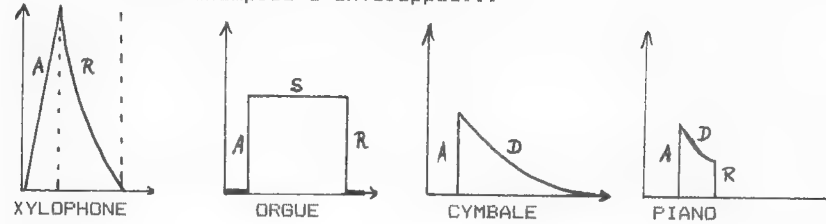
Voici deux enveloppes qui conviennent **au VIOLON:**



La note musicale est Soutenue au 2/3 de son volume maximal

La note mus. est Soutenue au niveau maximal de A

Voici 4 autres exemples d'enveloppes...



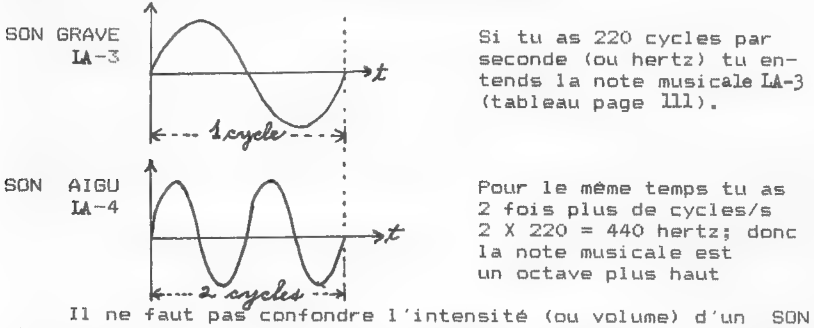
CONCLUSION:

L'Attaque, le Déclin, la Relaxation et la différence entre l'amplitude maximale (sommet) et le niveau de Soutien **sont des facteurs importants** pour différencier **le SON d'un** instrument **du SON d'un autre** instrument.

Cette étude de la 1ère qualité du SON (intensité) te permet de mieux comprendre les lignes 10 et 20 de l'exercice 41. Passe à la 2ième qualité du SON: la hauteur.

B- La HAUTEUR d'un SON

Parler de la **hauteur** d'un SON c'est parler de sa **fréquence.** La **hauteur** est cette qualité qui nous permet de distinguer un SON aigu d'un SON grave...Et ce qui distinque un SON grave d'un SON aigu c'est sa fréquence, c'est-à-dire le nombre de vibrations à la seconde (ou hertz)



Si tu as 220 cycles par seconde (ou hertz) tu entends la note musicale LA-3 (tableau page 111).

Pour le même temps tu as 2 fois plus de cycles/s 2 X 220 = 440 hertz; donc la note musicale est un octave plus haut

Il ne faut pas confondre l'intensité (ou volume) d'un SON et la hauteur (ou fréquence) de ce SON. Le LA-3 et le LA-4 ont la même intensité (ou volume ou amplitude). Par contre le LA-4 est 2 fois plus haut (440 Hz) que le LA-3 (220 Hz).

Je te signale que les êtres humains peuvent percevoir des SONS allant de 16 à 16000 vibrations par seconde. Ces limites varient d'un individu à l'autre...Un musicien peut entendre des fréquences plus basses et plus hautes que le commun des mortels! Les compositions musicales ont généralement des fréquences supérieures à 27 Hz et inférieures à 4700 Hz.

Les 8 octaves de ton COMMODORE 64

Tu connais déjà les 8 sons de la **gamme majeure:**
Do, Ré, Mi, Fa, Sol, La, Si, DO. La 1ère note s'appelle la tonique. La dernière note (DO) est l'octave aigue de la première, c'est-à-dire que sa fréquence est 2 fois plus grande que la première.

La **gamme tempérée** divise l'octave en **12 demi-tons égaux:**
Do, Do#, Ré, Ré#, Mi, Fa, Fa#, Sol, Sol#, La, La#, Si.

Chaque demi-ton vaut $\sqrt[12]{2}$ ou 2↑(1/12) c'est-à-dire: 1,05946.
Expérience: Regarde le tableau des fréquences page 112 et tu auras le La-4 avec sa fréquence 440 Hz **(Hertz)** . C'est la note-type qui fait l'uniformité entre la plupart des pays...C'est le **La du violon** (440 Hz).

Pour obtenir la fréquence du **La#-4** (# pour dièse) qui est un demi-ton plus haut, **tu multiplies 440 par 1,05946** ou bien 440 X 2 ↑ (1/12). (Sur l'ordinateur **tu** dois mettre un **point** pour séparer les entiers des décimales: 1.05946) **Fais ce calcul.** La note **La#-4 correspond à la note Si♭ -4**. Le signe ♭ pour bémol, abaisse la note d'un demi-ton.

Ton C64 peut t'offrir 8 octaves et comme chaque octave est divisée en 12 demi-tons, c'est une échelle de 96 notes (8X12=96) que tu peux obtenir. En pratique, tu t'arrêtes à 95 (0 à 94) comme dans l'exercice 45 ci-dessous.

<u>EXERCICE 45</u> Ce programme fera défiler 6 colonnes de nombres sur l'écran: 1-les Nos des notes (0à94) 2-les Notes+octaves 3- les hautes fréquences(HF) 4-les basses fréq. (BF) 5- les fréq. en décimal(FD) 6-les fréq. en hertz (Hz). Tu devras aussi obtenir un SON pour chacun des 12 demi-tons des 8 octaves(95 SONS). **Tourne le bouton volume TV**

```
5 REM*LES 8 OCTAVES DE LA GAMME TEMPEREE DES MUSICIENS*PAR L.PICARD
10 S=54272: A=15.434: REM A=15.091 POUR GAMME TEMP. DES PHYSICIENS
15 POKE S+24,15
20 POKE S+5,8: POKE S+6,0
25 PRINT"□NOS" TAB(5)"NOTES+" TAB(13)"HF" TAB(18)"BF" TAB(24)"FD";
26 PRINT TAB(31)"HERTZ" : PRINT TAB(5)"OCTAVES": PRINT
27 OC$="-0"
30 FOR I=0 TO 94
35 READ N$
36 IF N$="$$" THEN OC=OC-1: RESTORE: OC$=STR$(OC): READ N$: PRINT
40 POKE S+4,17
50 A=A*2↑(1/12): FD=INT(A/.059605)
60 HF=INT(FD/256): BF=FD-(HF*256)
70 POKE S+1,HF: POKE S,BF
80 FOR T=1 TO 300: NEXT
85 POKE S+4,16
90 PRINT I TAB(5)N$ OC$ TAB(12)HF TAB(17)BF TAB(23)FD;
91 PRINT TAB(30)INT(A*100+.005)/100
95 NEXT I
100 DATA DO,"DO#",RE,"RE#",MI,FA,"FA#",SOL,"SOL#",LA,"LA#",SI,"$$"
```

HF: Haute Fréquence
BF: Basse Fréquence
FD: Fréq. en décimal
HZ: Fréq. en hertz

NOS	NOTES+OCTAVES	HF	BF	FD	HZ
0	DO-0	1	18	274	16.35
1	DO#-0	1	34	290	17.32
2	RE-0	1	51	307	18.35
3	RE#-0	1	70	326	19.44
4	MI-0	1	89	345	20.6
5	FA-0	1	110	366	21.82
6	FA#-0	1	131	387	23.12
7	SOL-0	1	155	411	24.49
8	SOL#-0	1	179	435	25.95
9	LA-0	1	205	461	27.5
10	LA#-0	1	232	488	29.13
11	SI-0	2	5	517	30.86
12	DO-1	2	36	548	32.7
13	DO#-1	2	69	581	34.64
14	RE-1	2	103	615	36.7
15	RE#-1	2	140	652	38.89
16	MI-1	2	179	691	41.2
17	FA-1	2	220	732	43.65
18	FA#-1	3	7	775	46.24
19	SOL-1	3	54	822	48.99
20	SOL#-1	3	102	870	51.91
21	LA-1	3	154	922	55
22	LA#-1	3	209	977	58.27
23	SI-1	4	11	1035	61.73
24	DO-2	4	73	1097	65.4
25	DO#-2	4	138	1162	69.29
26	RE-2	4	207	1231	73.41
27	RE#-2	5	24	1304	77.78
28	MI-2	5	102	1382	82.4
29	FA-2	5	184	1464	87.3
30	FA#-2	6	15	1551	92.49
31	SOL-2	6	108	1644	97.99
32	SOL#-2	6	205	1741	103.82
33	LA-2	7	53	1845	110
34	LA#-2	7	163	1955	116.54
35	SI-2	8	23	2071	123.47
36	DO-3	8	146	2194	130.81
37	DO#-3	9	21	2325	138.59
38	RE-3	9	159	2463	146.83
39	RE#-3	10	49	2609	155.56
40	MI-3	10	205	2765	164.81
41	FA-3	11	113	2929	174.61
42	FA#-3	12	31	3103	184.99
43	SOL-3	12	216	3288	195.99
44	SOL#-3	13	155	3483	207.65
45	LA-3	14	107	3691	220
46	LA#-3	15	70	3910	233.08
47	SI-3	16	47	4143	246.94

NOS	NOTES+OCTAVES	HF	BF	FD	HZ
48	DO-4	17	37	4389	261.62
49	DO#-4	18	42	4650	277.18
50	RE-4	19	62	4926	293.66
51	RE#-4	20	99	5219	311.12
52	MI-4	21	154	5530	329.63
53	FA-4	22	227	5859	349.23
54	FA#-4	24	63	6207	369.99
55	SOL-4	25	176	6576	391.99
56	SOL#-4	27	55	6967	415.3
57	LA-4	28	214	7382	440
58	LA#-4	30	140	7820	466.16
59	SI-4	32	94	8286	493.88
60	DO-5	34	74	8778	523.25
61	DO#-5	36	84	9300	554.37
62	RE-5	38	125	9853	587.33
63	RE#-5	40	199	10439	622.25
64	MI-5	43	52	11060	659.26
65	FA-5	45	198	11718	698.46
66	FA#-5	48	126	12414	739.99
67	SOL-5	51	97	13153	783.99
68	SOL#-5	54	111	13935	830.61
69	LA-5	57	172	14764	880
70	LA#-5	61	25	15641	932.33
71	SI-5	64	188	16572	987.77
72	DO-6	68	149	17557	1046.51
73	DO#-6	72	169	18601	1108.74
74	RE-6	76	251	19707	1174.67
75	RE#-6	81	143	20879	1244.51
76	MI-6	86	105	22121	1318.52
77	FA-6	91	140	23436	1396.92
78	FA#-6	96	253	24829	1479.99
79	SOL-6	102	194	26306	1567.99
80	SOL#-6	108	222	27870	1661.23
81	LA-6	115	88	29528	1760.01
82	LA#-6	122	51	31283	1864.67
83	SI-6	129	120	33144	1975.55
84	DO-7	137	42	35114	2093.02
85	DO#-7	145	82	37202	2217.48
86	RE-7	153	247	39415	2349.34
87	RE#-7	163	30	41758	2489.03
88	MI-7	172	210	44242	2637.04
89	FA-7	183	24	46872	2793.85
90	FA#-7	193	251	49659	2959.98
91	SOL-7	205	132	52612	3135.99
92	SOL#-7	217	189	55741	3322.46
93	LA-7	230	176	59056	3520.03
94	LA#-7	244	103	62567	3729.34

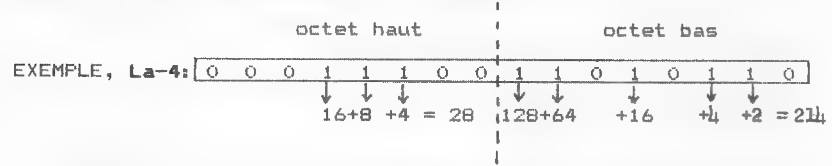
Tu peux remarquer, dans la colonne des hertz, que la note musicale "LA" donne un **nombre entier précis à chaque octave** et ce nombre **double d'une octave à l'autre:...55, 110, 220, 440... C'est parce que ce programme te donne la gamme tempérée des musiciens, basée sur le "LA" du violon: 440 Hz.**

NOTE: Tu remarques aussi que les valeurs obtenues pour la BF peuvent varier de + ou - 1 avec celles de ton "User's Guide"(p.152). Ne t'en fais pas avec ce "1" et prends les valeurs que te donne le programme ci-dessus car le tableau du "User's Guide"(p.152) a aussi la précision de + ou - 1.

Si tu veux obtenir la gamme tempérée **des physiciens** (p.384 dans "Reference Guide"), **mets A=15.091** à la ligne 10 du programme

Voici une **précision importante** concernant la haute frèquence(HF) et la basse fréq.(BF):
Avec un OCTET (huit bits) tu peux obtenir 256 valeurs différentes comprises entre 0 et 255. Si tous les bits son à 1 tu obtiens 255.

Or la valeur décimale des notes des 8 octaves peut prendre des valeurs comprises entre 0 et 65535. Pour obtenir ces valeurs il nous faudrait 16 bits...Ces 16 bits tu les obtiens si tu jumelles 2 octets (total: 16 bits). **L'octet de gauche s'appelera <u>l'octet haut</u>** et l'octet de droite **portera le nom d'octet bas.** Donc il faudra **2 registres (mémoires)** pour inscrire la valeur décimale des notes.



Regarde le **La-4 de ton tableau**(p.112) et tu verras:
HF = 28 et BF = 214
Donc la valeur de l'octet haut donne la HF et la valeur de l'octet bas donne la BF.

Mais attention! Pour avoir la valeur **décimale** de l'octet haut, il faut le **multiplier par 256.** La valeur de l'octet bas correspond déjà à sa valeur décimale. Donc les valeurs inscrites dans la colonne FD(fréquence en décimal) s'obtiennent ainsi:
Exemple, le La-4: 28(HF) multiplié par 256 + 214(BF) = 7382.

Conclusion: Cette étude de **la fréquence (ou hauteur du son)** te permet de mieux comprendre la **ligne 40 de l'exercice 41** où 2 registres apparaissent pour indiquer la fréq.: Pour la voix 1, tu mets la valeur de la haute fréq.(HF) dans le registre 54273 et la valeur de la basse fréq.(BF) dans le registre 54272.

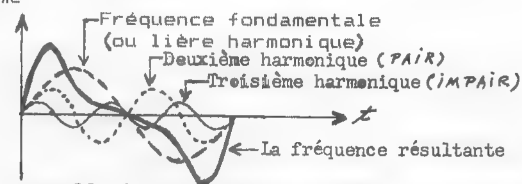
C- Le TIMBRE d'un SON

Tu fermes les yeux et tu écoutes la **même note musicale (même fréq.)** jouée **à l'orgue, au piano** puis **au violon**. Pourquoi peux-tu identifier l'instrument qui joue cette même note?

C'est parce que le **même SON, produit par des instruments différents a un TIMBRE différent**...Et le timbre d'une note musicale dépend de ses **HARMONIQUES**...Et l'ensemble des harmoniques **déterminent la FORME de l'ONDE.**

Inversement, si tu changes la FORME de l'ONDE, tu fais varier le TIMBRE(ou le nombre d'harmoniques). Nous verrons ci-dessous que ton C64 t'offre le choix entre 4 FORMES d'ONDES.

Il est intéressant de noter aussi qu'**un SON sans harmonique, comme celui du diapason**, est moins agréable à entendre qu'un SON possédant toutes les harmoniques comme celui fourni par un violon.



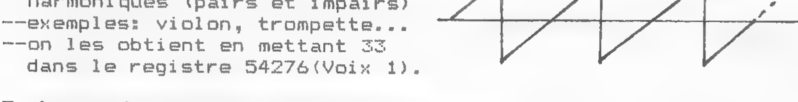
On appelle harmonique d'un SON, un autre son dont la fréquence est un multiple entier de la fréquence du premier.

Ton C64 peut produire 4 FORMES d'ONDES:

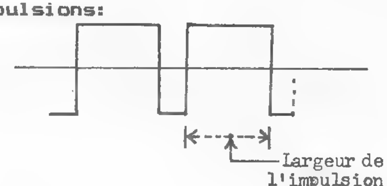
1- Les ondes triangulaires:
--elles possèdent seulement les harmoniques impairs
--exemples: orgue, flute...
--on les obtient en mettant 17 dans le registre 54276(Voix 1).



2- Les ondes à dents de scie:
--elles possèdent toutes les harmoniques (pairs et impairs)
--exemples: violon, trompette...
--on les obtient en mettant 33 dans le registre 54276(Voix 1).



3- Les ondes rectangulaires ou impulsions:
--elles possèdent seulement les harmoniques impairs
--exemple: piano...
--on les obtient en mettant 65 dans le registre 54276 (Voix 1)

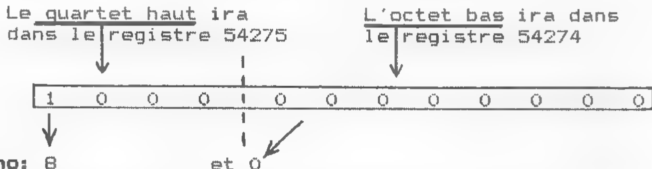


Ici il faut apporter une précision importante: Les impulsions(ou ondes rectang.) ont une largeur et **si tu fais varier cette largeur, tu changes le timbre** de la note musicale.

La largeur peut varier de 0 à 4095; à ces 2 extrêmes l'impulsion devient une ligne droite (tension continue).

Dans le cas des impulsions comme dans celui des fréquences, il faudra loger l'information dans 2 registres puisque la plus grande valeur dépasse 255! Tu auras **un quartet(4 bits) pour la haute impulsion(HP)** et **un octet pour la basse impulsion(BP). Le maximum sera: quartet 15X256 + l'octet 255 = 4095.**

NOTE: la valeur médiane (2048) pour la largeur de
l'impulsion est l'idéal pour le timbre du piano
parce que 2048 donne une onde carrée: largeur=hauteur.



Exemple, piano: 8 et 0

En résumé, pour avoir une onde rectangulaire il faut: **a) mettre 65 dans le registre 54276** et **b) ajouter 2 instructions pour préciser la haute impulsion(HP) et la basse impulsion(BP). Tu écriras: POKE 54275,8: POKE 54274,0 (pour le piano, voix 1).**

Voici 2 **remarques importantes** avant de faire l'exercice 46:
a) Dans un programme, **l'instruction donnant la FORME de l'ONDE** doit toujours être **placée après** les instructions précisant **l'enveloppe de l'onde.**
b) **Lorsque le bit zéro** du régistre 54276 **est mis à zéro**(OFF), c'est-à-dire quand 17 devient 16 ou 33 devient 32 ou 65 devient 64 ou 129 devient 128, **ALORS l'enveloppe n'est plus activée** et la **phase de relaxation commence.**

EXERCICE 46 Fais le chargement(LOAD) de l'exercice 41-B. A la ligne 20, mets 9 et 0 dans les registres S+5 et S+6. A la ligne 50, mets 65 au lieu de 17 pour avoir une **impulsion** et à la ligne 70, remplace 16 par 64. **Quand** tu as une impulsion, il faut 2 instructructions supplémentaires: ajoute une ligne No 25 et mets 8 pour la haute impulsion(HP) et 0 pour la basse impulsion(BP); tu auras 8X256=**2048 (onde carrée pour piano):**

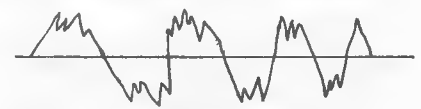
Solution au verso:

```
5 S=54272 :   REM  PIANO
10 POKE S+24,15
20 POKE S+5,9 : POKE S+6,0
25 POKE S+3,8: POKE S+2,0
30 READ HF,BF,DU: IF HF=-1 THEN END
40 POKE S+1,HF: POKE S,BF
50 POKE S+4,65
60 FOR T=1 TO DU: NEXT
70 POKE S+4,64: FOR T=1 TO 20:NEXT
80 GOTO 30
500 DATA 21,31,512,25,30,384,25,30,128,25,30,512
510 DATA 28,49,512,31,165,384,31,165,128,31,165,768
520 DATA 25,30,256,28,49,256,31,165,256,33,135,512
530 DATA 23,181,512,28,49,512,25,30,512
540 DATA -1,-1,-1
```

4- Le BRUIT (<u>ou les ONDES discordantes</u>):

La 4ième FORME d'onde est celle du **BRUIT.** Si tu mets la valeur 129 dans le registre 54276, le C64 fournira un mélange d'ondes choisies au hasard et comme ces ondes ont des fréquences qui ne sont pas des multiples entiers...(pas d'harmoniques), il en résulte des effets discordants (BRUITS).

Exemples: tonnerre, p. 59
moteurs(fusée),p.65
le vent, les pas...



CONCLUSION: Le TIMBRE du SON rendait nécessaire une étude sur les différentes FORMES d'ondes de ton synthétiseur de musique **car la forme de l'onde permet de changer le timbre d'un son.** Voici un exercice qui résume les notions apprises sur la programmation de la musique...

<u>**EXERCICE 47**</u> Ce programme permet d'entendre la chanson populaire "A LA CLAIRE FONTAINE" jouée sur différents instruments de ton choix:

```
10 PRINT"■■"SPC(12)"A LA CLAIRE FONTAINE"
15 PRINT" ■■1.■ PIANO
17 PRINT" ■■2.■ VIOLON
19 PRINT" ■■3.■ TROMPETTE
21 PRINT" ■■4.■ FLUTE
23 PRINT" ■■5.■ ORGUE
25 PRINT" ■■6.■ XYLOPHONE
27 PRINT" ■■7.■ SYNTHETISEUR
28 PRINT"■■TAPE UN ■CHIFFRE■ ET ■RETURN
30 INPUT"■■TON CHOIX";CH
40 S=54272: FOR M=S TO S+24: POKE M,0: NEXT
50 ON CH GOTO 60,70,80,90,100,110,120
60 AD=9: SR=0: FO=65: HP=8: BP=0: B0=64: GOTO 130: REM PIANO
70 AD=168: SR=169: FO=33: B0=32: GOTO 130: REM VIOLON
80 AD=84: SR=128: FO=33: B0=32: GOTO 130: REM TROMPETTE
90 AD=96: SR=128: FO=17: B0=16: GOTO 130: REM FLUTE
100 AD=0: SR=240: FO=17: B0=16: GOTO 130: REM ORGUE
110 AD=26: SR=5: FO=17: B0=16: GOTO 130: REM XYLOPHONE
```

```
120 AD=144: SR=243: FO=33: B0=32: GOTO 130: REM UNIQUE AU SYNTHETISEU
130 POKE S+24,15
140 READ HF,BF,DU
150 IF HF=-1 THEN RESTORE: GOTO 10
160 POKE S+5,AD: POKE S+6,SR
170 POKE S+3,HP: POKE S+2,BP
180 POKE S+4,FO
190 POKE S+1,HF: POKE S,BF
200 FOR T=1 TO DU: NEXT
210 POKE S+4,B0: FOR T=1 TO 40: NEXT
220 GOTO 140
500 DATA 25,177,512,25,177,256,32,94,256,32,94,256
510 DATA 28,214,256,32,94,256,28,214,256,25,177,512
520 DATA 25,177,256,32,94,256,32,94,256,28,214,256
530 DATA 32,94,768,32,94,512,32,94,256,28,214,256
540 DATA 25,177,256,32,94,256,38,126,256,32,94,256
550 DATA 38,126,512,38,126,256,32,94,256,25,177,256
560 DATA 32,94,256,28,214,768,25,177,512,25,177,256
570 DATA 32,94,256,32,94,256,28,214,128,25,177,128
580 DATA 32,94,256,25,177,256,32,94,512,32,94,256
590 DATA 28,214,128,25,177,128,32,94,256,28,214,256,25,177,768
600 DATA -1,-1,-1
```

L'instructions **ON** (ligne50) sera étudiée au chapitre 4.

Je te conseille d'acheter le recueil des "**Cent plus belles chansons**" et d'en faire jouer par ton C64...Ce recueil contient **les chansons les plus populaires de notre folklore québécois.** Les 2 exemples précédents sont dans ce recueil: **MON BEAU SAPIN et A LA CLAIRE FONTAINE**...tu pourras comparer...

Dans le projet "QUEBEC CHANTE", ton C64 jouera la chanson de Gilles Vigneault: "Gens du pays..."

EXERCICE 48-A Ecris ce programme qui fera entendre 5 fois les notes La-4 et Do-5 avec un délai de 1/3 seconde entre chaque note.

```
5 S=54272
10 FOR I=1 TO 5
20 POKE S+24,15
30 POKE S+5,0: POKE S+6,240
35 POKE S+4,17
40 POKE S+1,28: POKE S,214
45 FOR T=1 TO 300: NEXT
50 POKE S+1,34: POKE S,74
60 FOR T=1 TO 300: NEXT
70 POKE S+4,16
80 NEXT I: POKE S+1,0: POKE S,0
```

EXERCICE 48-B Pour créer un **effet de rapprochement** tu fais augmenter graduellement le volume. Transforme ainsi les lignes 10 et 20:

```
10 FOR I=1 TO 15 STEP 1.5
20 POKE S+24,I
```

EXERCICE 49 Pour créer un **effet d'éloignement** il faut diminuer graduellement le volume. Complète ton programme pour obtenir successivement les 2 effets:

```
5 S=54272
10 FOR I=1 TO 15 STEP 1.5: GOTO 20
15 FOR I=15 TO 0 STEP -1.5
20 POKE S+24,I
30 POKE S+5,0: POKE S+6,240
35 POKE S+4,17
40 POKE S+1,28: POKE S,214
45 FOR T=1 TO 300: NEXT
50 POKE S+1,34: POKE S,74
60 FOR T=1 TO 300: NEXT
70 POKE S+4,16
75 IF I=0 THEN END
80 NEXT I: POKE S+1,0: POKE S,0
90 GOTO 15
```

EXERCICE 50 Ce programme fait rebondir une balle retenue par un élastique. **Un son électronique est produit** lorsque la balle touche le sol et un autre de fréquence supérieure quand la balle touche la palette.

Listing du programme | Dessin sur l'écran

```
5 PRINT"▯▮▮▮▮▮▮▮": POKE 53281,0
10 S=54272: POKE S+24,15: POKE S+5,8: POKE S+6,0
20 PRINT"       ▮ ╮
21 PRINT"       ▮ o ▬
22 PRINT"       ◣ ▮◤
23 PRINT"       ╱▮
24 PRINT"       ▮||
25 PRINT"        ||
26 PRINT"       ▮┌─────▮
40 FOR J=1 TO 10
45 X=1353: Z=0: B=81: E=93
50 POKE X,B: FOR T=1 TO 50: NEXT
55 FOR I=1 TO 3
60 Z=Z+40
65 POKE X+Z,B: POKE X+Z-40,E: FOR T=1 TO 50: NEXT
70 IF I=3 THEN POKE S+1,25: POKE S,177: POKE S+4,17
71 POKE S+1,0: POKE S,0: POKE S+4,16
75 IF C=1 THEN END
80 NEXT I: Z=160
85 POKE X-40,98: FOR T=1 TO 10: NEXT: POKE X-40,111
90 FOR I=1 TO 3
95 Z=Z-40
100 POKE X+Z,32: POKEX+Z-40,81: FOR T=1 TO 50: NEXT
105 IF I=3 THEN POKE S+1,51: POKE S,97: POKE S+4,17
106 POKE S+1,0: POKE S,0: POKE S+4,16
110 NEXT I
115 POKE X-40,98: FOR T=1 TO 15: NEXT: POKEX-40,111
120 NEXT J
125 C=1: GOTO40
```

PROJET D-2 Québec chante

Prends ta cassette sur laquelle tu conserves les programmes déjà étudiés et fais le chargement(LOAD) du projet D-1 (page 30) dans ton C64. Après RUN et RETURN la silhouette du Québec apparaîtra sur l'écran.

1- Ecris LIST et introduis, à droite de la silhouette du Québec, la phrase
   suivante: Mon cher
             Québec
             C'est à
             Ton tour

2- Remplace la boucle infinie de la ligne 1230 par une boucle d'attente (ou de délai) de 1 1/2 secondes.

3- Procure-toi une copie du chant populaire de Gilles Vigneault "Gens du Pays"...Tu constates qu'il y a seulement 6 notes différentes: FA, SOL, LA, LA#, DO, RE.

4- Utilise la VOIX 1 (registres 53272 à 53278: p.122) du C64 et place la valeur des notes musicales dans des instructions DATA (voir tableau des notes page 111+112). Mets le volume principal à 15.

5- Ecris la basse et la haute fréquence des notes selon l'ordre du chant sans oublier d'inclure <u>les</u> temps de délai(DUrée) dans les instructions DATA.

6- Utilise une boucle FOR...NEXT pour lire(READ) <u>trois données</u> à chaque itération: HF,BF, et DU (<u>temps de délai</u>.

7- Il serait bon d'inclure une instruction GET pour permettre à l'utilisateur de réentendre le chant simplement en appuyant sur une touche. Dans ce cas, l'instruction RESTORE deviendra nécessaire pour ramener le pointeur au début du premier DATA (donnée).

8- Tu utiliseras la variable S pour identifier le premier registre du SON (Voix 1:S=53272)

Une solution de ce projet D-2 est proposée à la page suivante.

PROGRAMME D-2: Québec chante

```
990 PRINT"□"
1000 PRINT
1010 PRINT"
1020 PRINT"
1030 PRINT"
1040 PRINT"                                MON CHER"
1050 PRINT"                                  QUEBEC"
1060 PRINT"                                 C'EST"
1070 PRINT"                                 A TON"
1080 PRINT"                                  TOUR"
1090 PRINT"
1100 PRINT"
1110 PRINT"
1120 PRINT"
1130 PRINT"
1140 PRINT"
1150 PRINT"
1160 PRINT"
1170 PRINT"
1180 PRINT"
1190 PRINT"
1200 PRINT"
1210 PRINT"
1230 FORT=1TO1500:NEXT
2010 S=54272
2020 POKE S+24,15: REM VOLUME MAX.
2025 POKE S+3,0: POKE S+2,255: REM OCTET HT ET BAS DE L'IMPULSION
2030 POKE S+5,9: POKE S+6,9: REM AD ET SR DE LA VOIX 1
2040 READ HF,BF,DU
2050 IF HF<0 THEN 2110
2060 POKE S+1,HF: POKE S,BF: REM METTRE LA HTE FREQ. ET BASSE FREQ.
2070 POKE S+4,65: REM BIT ZERO A 1 DE L'ONDE RECTANGULAIRE
2080 FOR T=1 TO DU: NEXT
2090 POKE S+4,64: FOR T=1 TO 50: NEXT: REM BIT ZERO A 0 DE L'ONDE RECT.
2100 GOTO 2040
2110 PRINT"TAPE UNE TOUCHE";
2120 GETM$:IFM$=""THEN 2120
2130 RESTORE:GOTO990
2500 DATA 28,214,256,25,177,256,22,227,256,34,75,768
2510 DATA 28,214,256,25,177,256,22,227,256,38,126,768
2520 DATA 25,177,256,30,141,256,38,126,256,34,75,512
2530 DATA 34,75,256,34,75,512,38,126,256,34,75,512
2535 REM ************
2540 DATA 28,214,256,25,177,256,22,227,256,34,75,768
2550 DATA 28,214,256,25,177,256,22,227,256,38,126,768
2560 DATA 25,177,256,30,141,256,38,126,256,34,75,512
2570 DATA 22,227,256,28,214,512,25,177,256,22,227,768,-1,-1,-1
```

Le rôle de chacune des lignes est expliqué à la p.121

Rôle de chacune des lignes du programme D-2 de la page 120

990 à 1230: Si tu veux des précisions sur ces lignes, relis la page 30.

2010: Identification de la variable S (S=1er registre du SID).

2020: Volume principal à maximum (15).

2025: Les 2 registres de la haute et la basse impulsion qu'il faut employer avec l'onde rectangulaire.

2030: L'enveloppe: 9 pour Attaque+Déclin et 9 pour Soutien+Relaxation.

2040: Lecture des 3 premières valeurs dans DATA de la ligne

2500: HF pour la haute fréqence, BF pour la basse fréq. et DU pour le délai

2050: Offrir le choix de recommencer quand le C64 rencontre -1 dans 'DATA'.

2060: A la première itération la haute fréq.(HF) vaut 28, la basse fréq.(BF) vaut 214 et la durée(DU) 256 ms (ou 1/4 seconde)

2070: Avec 65 dans le registre 53276 tu as une onde rectangulaire et tu **actives** le registre de l'enveloppe (bit zéro à 1).

2080: Boucle d'attente...

2090: Avec la valeur 64 l'activation de l'enveloppe cesse (bit zéro à 0).

2100: Recommencer la boucle pour lire(READ) 3 autres données...

2120: Attente d'un caractère quelconque.

2130: L'instruction RESTORE ramène le pointeur des données (DATA) au début (en 2500)...pour permettre de relire les valeurs: HF,BF et DU.

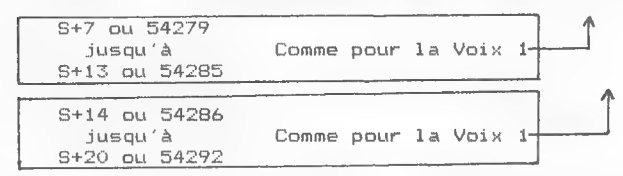
Note	HF	BF	Mot	Durée
LA	28	214	Mon	256
SOL	25	177	cher	256
FA	22	227	Qué-	256
DO	34	75	bec	768
LA	28	214	c'est	256
SOL	25	177	à	256
FA	22	227	ton	256
RE	38	126	tour	768
SOL	25	177	de	256
LA#	30	141	te	256
RE	38	126	lais-	256
DO	34	75	ser	512
DO	34	75	par-	256
DO	34	75	ler	512
RE	38	126	d'a-	256
DO	34	75	mour	512

Le chant reprend...et se termine ainsi:

FA	22	227	par-	256
LA	28	214	ler	512
SOL	25	177	d'a-	256
FA	22	227	mour	768

Les 29 registres (mémoires) du synthétiseur de son/musique sont situés aux **adresses 54272 à 54300**. Le **symbole S** sera la variable **servant à représenter la première adresse:** S=54272.

Les registres ou adresses du SID	Le contenu des adresses	Rôle du registre	
S = 54272	0 à 255	Basse fréquence (BF)	
S+1 ou 54273	0 à 255	Haute fréquence (HF)	
S + 2	0 à 255	Basse impulsion(BP)	ONDES RECT.
S + 3	0 à 15	Haute impulsion(HP)	
S + 4	17ou33ou65ou129	Forme de l'onde (FO)	
S + 5 →	A=0 à 15 fois16 D=0 à 15	Attaque/Déclin (AD)	
S+6 ou 54278 →	S=0 à 15 fois16 R=0 à 15	Soutien/Relaxat.(SR)	



Les adresses S+21, S+22, et S+23 sont les registres des **FILTRES.** C'est une partie importante de ton synthétiseur qui sera expérimentée dans le VOLUME 2 de cette série. Les filtres peuvent changer le timbre comme peut le faire "la forme de l'onde" et encore davantage!
S+24 ou 54296 est l'adresse du volume principal pour **les 3 VOIX**
S+25 à S+29 sont des registres (en lecture seul.) que tu expérimenteras également dans le volume 2.

Tu peux programmer **une pièce musicale ayant plusieurs parties**...Mais pour synchroniser l'ensemble **il est bon d'utiliser les tableaux à 2 dimensions** (voir chap.5). C'est principalement pour cette raison que les programmes des chansons en plusieurs parties se retrouvent dans le deuxième volume! Tu comprendras **aussi** l'avantage d'utiliser la forme décimale (FD) dans les "DATA" au lieu de 2 valeurs: HF et BF.

Conclusion: Je te conseille de **conserver dans un tableau** les meilleures combinaisons: **AD SR** et **FORME de l'ONDE** que tu auras découvertes dans tes expériences musicales...**EXEMPLE:**

A + D : AD	**S + R : SR**	**Forme de l'onde: 17ou33ou65ou129 Si FO = 65 alors préciser HP et BP**	**Instru-ments**
0 + 0 = 0	240+ 0 = 240	**17**	Orgue
160+ 8 = 168	160+ 9 = 169	**33**	Violon
0 + 9 = 9	0 + 0 = 0	**65** HP=8 BP=0	Piano
etc...			

CHAPITRE IV

ACTIVITES DIRIGEES Série No 4

BUT Cette quatrième série d'activités a pour but de te familiariser avec:

1- Les 8 instructions: INPUT, GOSUB et RETURN, STOP, END, DIM, ON et LET.

2- Les boucles imbriquées.

3- La notion d'algorithme.

4- Les six fonctions: SQR, INT, STR$, VAL, RND.

5- Les opérateurs arithmétiques.

6- Les tableaux à une dimension (ou listes). Les variables indicées.

7- Un "didacticiel".

PROJET E-1

Dessine un circuit électrique sur l'écran. Entre ton dessin dans un programme par la méthode des PRINT. Sauve ton programme sur cassette ou sur disque car tu en auras besoin au chapitre IV.

Le circuit doit contenir:

1- Un accumulateur de 12 volts.

2- Un interrupteur.

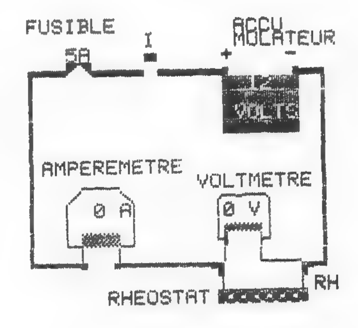
3- Un fusible de 5 ampères.

4- Un volmètre

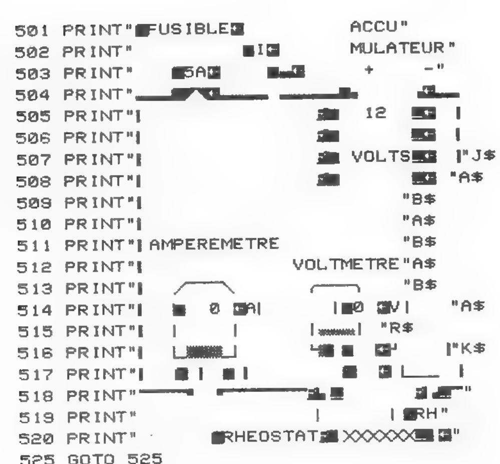
5- Un ampèremètre.

6- Un rhéostat (résistance variable).

Voici un exemple de ce que tu pourrais obtenir sur l'écran après 'RUN et RETURN'.



LISTING du PROGRAMME E-1:



INPUT

L'instruction INPUT (entrer) permet d'entrer, par le clavier, des données numériques ou des données chaînes de caractères dans la mémoire de l'ordinateur pendant l'exécution d'un programme.

INPUT est donc une instruction d'entrée de données en mode conversationnel.

PREMIER EXEMPLE: Tu veux un programme qui calcule:

1- la circonférence C d'un cercle. **Formule:** $C = 2\pi r$
2- la surface S de ce cercle. **Formule:** $S = \pi r^2$

L'ordinateur attendra une seule donnée numérique: le rayon r du cercle.

EXERCICE 1 Ecris une première version d'un programme qui calcule la circonférence et la surface d'un cercle.

```
10 INPUT R
20 C=2*π*R
30 S=π*R↑2
40 PRINT C,S
```

L'instruction INPUT (ligne 10) **interrompt** l'exécution du programme et fait apparaître un point d'interrogation sur l'écran pour indiquer que l'ordinateur attend une donnée: le rayon R d'un cercle.

Admettons que tu tapes 15 et que tu appuies RETURN. Aussitôt le C64 met cette valeur 15 dans la variable R qui représente le rayon.

A la ligne 20 le C64 **calcule** la valeur de **la circonférence** et met le **résultat dans** la **variable C.**

A la ligne 30 il **calcule** la **surface** et met le **résultat dans** la **variable S.**

Ces calculs se font sans difficulté parce que le C64 a "toujours en mémoire" la valeur de π (3,14159...).

* symbole de la multiplication
↑ symbole de l'élévation à la puissance (ou exponentiation)

NOTE π*R↑2 est équivalent à π*R*R

A la ligne 40 le C64 **affiche** les **deux résultats** sur l'écran, c'est-à-dire le **contenu des variables C et S.**

Cette première version du programme fonctionne très bien

mais il serait bon de lui apporter les 4 améliorations suivantes:

1- C'est toi qui es l'auteur de ce programme et tu sais ce que l'ordinateur attend quand il affiche un ? . Il faudrait **ajouter une information pour qu'un autre utilisateur puisse connaître la question** que l'ordinateur lui pose. Cette information peut être écrite entre guillemets après INPUT. Tu ajoutes ensuite ; et la variable R. Ce ; placé entre l'information et la variable est indispensable.

EXERCICE 2 Remplace l'instruction 10 par:

```
10 INPUT"RAYON DU CERCLE"; R
```

Ecris RUN et tape RETURN. L'écran affiche:

```
RAYON du CERCLE?
```

2- Deux résultats seront affichés sur l'écran: celui de la circonférence et celui de la surface.

La aussi une information est nécessaire **pour que l'utilisateur différencie les 2 résultats:** celui de la circonférence et celui de la surface.

EXERCICE 3 Remplace l'instruction 10 par:

```
40 PRINT"CIRCONFERENCE =";C:PRINT"SURFACE =";S
```

3- Il reste encore une amélioration à apporter pour que l'échange homme-machine se fasse avec plus de rapidité et de confort.

Pour chaque valeur R entrée au clavier tu dois écrire RUN... Cette opération peut être éliminée.

EXERCICE 4 Utilise l'instruction GOTO pour ajouter une boucle infinie dans ton programme.

```
50 GOTO 10
```

Chaque fois que le C64 rencontre l'instruction 50, il retourne à la ligne 10 et redemande un nouveau RAYON. En principe, il est interdit d'implanter une boucle infinie dans un programme mais ici la boucle contient une instruction INPUT qui interrompt le programme. Tu peux alors **sortir du programme** quant tu le désires **si tu appuies simultanément les touches STOP et RESTORE.**

4- Avant de sauvegarder ton petit programme sur cassette, il serait bon d'améliorer l'affichage sur l'écran en te servant des caractères de COMMANDE d'AFFICHAGE (page 35).

EXERCICE 5 Ecris, en mode direct, deux instructions différentes pour effacer l'écran:

```
PRINT"♥"          OU          PRINT CHR$(147)
```

Mets l'une ou l'autre de ces instructions dans une ligne no 5 de ton programme.

EXERCICE 6 Ajoute le caractère No 5 p.35 au début de chacune des informations écrites entre guillemets et tu obtiendras cette **version finale** du programme:

```
1 REM CALCUL DE LA CIRCONFERENCE ET DE LA SURFACE D'UN CERCLE
5 PRINT"♥
10 INPUT"▓▓RAYON DU CERCLE";R
20 C=2*π*R
30 S=π*R↑2
40 PRINT"▓CIRCONFERENCE=";C: PRINT"▓SURFACE=";S
50 GOTO10
```

EXERCICE 7 Si tu veux garder seulement 2 décimales, il faut utiliser la fonction INT( )...(voir **exercice 19**). Remplace les lignes 20 et 30 par:

```
20 C=INT(2*π*R*100+.5)/100
30 S=INT(π*R↑2*100+.5)/100
```

DEUXIEME EXEMPLE:

On peut entrer plusieurs données dans une même instruction INPUT. Voici un exemple où le C64 **attend deux données:**

1- Tu veux un programme qui calcule la surface d'un triangle rectangle.

Formule: Surface = $\frac{\text{Base X Hauteur}}{2}$

L'ordinateur attendra deux données numériques: la base B et la hauteur H.

EXERCICE 8 Ecris une première version d'un programme qui calcule la surface d'un triangle rectangle:

```
10 INPUT B,H
20 S=B*H/2
30 PRINT S
```

(Note:le trait oblique est le symbole de la division).

Quand il y a plusieurs variables après INPUT, il faut les **séparer par une virgule.** Quand tu entres tes nombres, il faut aussi les séparer par une virgule.

Supposons que tu tapes 27,16 et que tu appuies RETURN. Aussitôt le C64 introduit la valeur 27 dans la variable B (base) et la valeur 16 dans la variable H (hauteur).

A la ligne 20, il fait le calcul et met la réponse dans la variable S (surface) puis à la ligne 30, il affiche la réponse (le contenu de S) sur l'écran.

QUESTION Si j'entre seulement un nombre, par exemple 27, et par erreur j'appuie RETURN. Que se passe-t-il?

Réponse: Ce n'est pas grave!... Le C64 prendra cette première valeur pour la mettre dans B puis il va te demander le deuxième nombre en affichant un double point d'interrogation.

Si tu tapes 3 valeurs au lieu de 2 le C64 prendra les deux premières et ignorera la troisième. Il imprimera: EXTRA IGNORED.

EXERCICE 9 Complète ton programme de la surface d'un triangle rectangle pour arriver à une version finale qui ressemblerait à celle du programme précédent(ex.6+7)

NOTE Si tu veux un programme qui calcule la surface d'un triangle quelconque, tu trouveras la réponse à l'exercice 18 de ce chapitre.

TROISIEME EXEMPLE:

EXERCICE 10 Mets une "variable chaîne" après INPUT et entre les données "chaînes de caractères" suivantes: (voir chaînes de caractères page 44).

```
10 INPUT"ECRIS TON PRENOM";A$
20 PRINT"CREME GLACEE MOLLE"
30 PRINT"P=PETIT FORMAT 0,35$"
40 PRINT"M=MOYEN    "CHR$(34)"    0,50$"
50 PRINT"G=GRAND    "CHR$(34)"    0,75$"
60 INPUT"TON CHOIX? P  M OU G";B$
70 IF B$="P" THEN P$="0,35$":GOTO100
80 IF B$="M" THEN P$="0,50$":GOTO100
90 P$="0,75$"
100 PRINT"CE SERA "P$" S.V.P."
110 PRINT"MERCI "A$
```

Explications:

A la ligne 10 le déroulement du programme est interrompu et l'ordinateur attend une chaîne de caractères: **ton prénom.** Il n'est pas nécessaire d'écrire to prénom entre guillemets car le C64 s'attend à une chaîne...

Quand ton prénom est écrit, tu appuies RETURN et le c64 met aussitôt ton prénom en mémoire à l'adresse de la "variable chaîne" A$.

Même chose à la ligne 45 sauf que la "variable chaîne" B$

ne contiendra qu'un caractère: soit P ou M ou G.

Lignes 30 et 35: Il arrive parfois qu'on utilise le caractère guillemet " **sous** un mot pour signifier la répétition de ce mot. Exemple:

```
Petit format   0,35$
Moyen    "     0,50$
Grand    "     0,75$
```

Le seul moyen d'introduire le caractère " dans une chaîne est l'emploi de la fonction CHR$(34)... (voir tableau des codes CHR$ p. 76).

La ligne 65 contrôle la réponse donnée à la ligne 60 car il est indispensable **qu'un des** 3 caractères: P ou M ou G soit tapé au clavier... Sinon redemander l'instruction 60.

BOUCLES IMBRIQUEES: (anglais: Nested Loops)

Tu as souvent utilisé les boucles simples FOR... NEXT dans les exercices précédents.

Une boucle imbriquée (ou intérieure) est une boucle contenue dans une autre boucle. Dans plusieurs programmes tu as placé une boucle d'attente (ou de délai); on peut dire que cette boucle d'attente était imbriquée lorsqu'elle était placée à l'intérieur d'une autre boucle. **Vérifie l'exemple suivant:**

Départ de ta fusée, lignes nos 1390 à 1420, page 65.

Construis maintenant un petit programme où la boucle imbriquée ne sera plus une simple boucle d'attente mais une boucle qui contiendra une instruction PRINT:

EXERCICE 11 Utilise 2 boucles imbriquées pour imprimer alternativement la devise: "Je me souviens" (en blanc) et 40 étoiles (bleu clair); répéter l'opération 11 fois.

```
5 PRINT"▯";
10 FOR I=1 TO 11
20 PRINT"▮JE ME SOUVIENS"
30 FOR J=1 TO 40
40 PRINT"▯*";
50 NEXT J
60 NEXT I
```

Il faut noter qu'on peut omettre les variables **J et I après** NEXT. Il est permis également de remplacer les deux instructions NEXT des lignes 50 et 60 par **une seule ligne écrite ainsi: 50 NEXT J,I.**

EXERCICE 12 Utilise 3 boucles imbriquées pour imprimer d'abord 5 traits verticaux (sur les colonnes 6, 13, 20, 27 et sur la colonne 34) et ensuite 39 traits horizontaux; cette opération doit se répéter 11 fois (n'oublie pas de compter la colonne 0).

```
5 PRINT"␣";
10 FOR I=1 TO 11
20 FOR J=6 TO 34 STEP 7
30 PRINT TAB(J)"|";
40 NEXT J
45 PRINT
50 FOR K=1 TO 39
60 PRINT"-";
70 NEXT K
75 PRINT
80 NEXT I
```

La boucle imbriquée FOR J... NEXT J sera parcourue 5 fois et J sera d'abord égal à 6 et ensuite à 13... à cause du PAS (STEP) 7. NOTE: la colonne 6 correspond à la 7e colonne (0 à 6) et la colonne 13 correspond à la 14e colonne (0 à 13).

La valeur de J sert de paramètre (argument) à la fonction TAB pour que la TABulation se fasse à l'endroit souhaité sur la rangée: colonnes 6, 13, 20...

L'étude des tableaux à 2 dimensions (chap. 5) t'offrira une occasion en or de pratiquer ces boucles imbriquées.

Règles à suivre dans les boucles imbriquées:

1- La boucle doit avoir une variable de contrôle différente de celles des autres boucles dans lesquelles elle est imbriquée
2- La boucle intérieure doit être entièrement contenue dans la boucle extérieure (pas de chevauchement).
3- On peut effectuer un saut à partir d'une boucle intérieure mais on ne peut pas effectuer un saut de l'extérieur vers l'intérieur... Tu ne peux rentrer que par une autre instruction FOR... NEXT.

OPERATEURS ARITHMETIQUES:

Il y a 5 opérateurs arithmétiques:

Addition	+	signe plus
Soustraction	-	signe moins
Multiplication	*	astérisque
Division	/	trait oblique (slash)
Exponentiation	↑	flèche verticale

Hiérarchie des opérations: (ou règles de préséance)

1- Toutes les exponentiations sont exécutées en premier.

2- Ensuite les multiplications ou les divisions sont effectuées en allant de la gauche vers la droite. (même hiérarchie)
3- Les additions ou les soustractions sont effectuées aussi en allant de la gauche vers la droite. (même hiérarchie)

NOTE Pour modifier la hiérarchie des opérations, on utilise les parenthèses car les parenthèses sont évaluées avec une priorité supérieure à toutes les autres opérations. D'abord les parenthèses les plus à l'intérieur... etc.

EXERCICE 13 Transformation des degrés Fahrenheit **F** en degrés Celcius **C**. **Formule:** C=(F-32)*5/9. On a introduit les parenthèses pour donner la priorité à la soustrac.:F-32

```
5 PRINT"■
10 INPUT"■■DEGRES FAHRENHEIT";F
20 C=(F-32)*5/9
30 PRINT:PRINT F"DEGRES FAHRENHEIT CORRESPOND A"
40 PRINT:PRINT C"DEGRES CELCIUS
50 GOTO 10
```

NOTION d'ALGORITHME

Le langage BASIC est un langage algorithmique. Ce mot algorithme te semble peut-être rébarbatif mais il faut quand même lire cette page et la suivante car **cette notion d'algorithme dépasse en importance le langage BASIC** avec lequel tu te familiarises présentement.

La première tache du programmeur consiste à exprimer, sous forme d'algorithme, le problème qu'on lui demande de résoudre. C'est seulement après cette première étape que le problème sera traduit sous forme de programme dans un langage donné: soit le BASIC, le PASCAL, le FORTRAN, le COBOL... etc.

Définition:

Un algorithme est une suite d'actions à effectuer pour résoudre un problème. Cette définition très brève permet de comparer un algorithme au mode d'emploi pour l'utilisation d'un appareil.

Par exemple, si tu veux téléphoner à un ami à partir d'une cabine téléphonique, tu dois:

1- avoir une pièce de monnaie
2- soulever l'écouteur (le combiné)
3- attendre la tonalité
4- insérer la pièce dans le boîtier
5- composer le numéro de ton ami
6- si...pas de réponse après avoir sonné 5 fois, raccrocher, récupérer ta monnaie, rappeler plus tard.
7- si...réponse, etc...

On peut considérer ce mode d'emploi, 1 à 5, comme la description de l'algorithme <u>d'utilisation</u> d'un téléphone public.

A la page 65 du "Programme d'études" 16-0081 du Ministère de l'Education, le mot ALGORITHME est défini ainsi: "Ensemble fini de règles déterminées servant à résoudre un problème au moyen d'un nombre fini d'opérations" (AFNOR).

Un exemple d'algorithme:

Le problème: Préparer un programme qui calcule les racines d'une équation du second degré.

Algorithme:

les données: soit $ax^2 + bx + c = 0$

On veut calculer les racines x_1 et x_2 au moyen des formules classiques:

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \qquad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

1- Entrer les données a, b, c au clavier

Les calculs et résultats:

2- Si a = 0 et b = 0 et c = 0, alors imprimer INDETERMINE
FIN (END)

3- Si a = 0 et b = 0, alors imprimer IMPOSSIBLE
FIN

4- Si a = 0, alors imprimer EQUATION DU PREMIER DEGRE
$x = -c/b$ Imprimer x
FIN

5- Soit le discriminant $D = b^2 - 4ac$
Si $D < 0$, alors imprimer PAS DE RACINES REELLES
FIN

6- Pour avoir des racines réelles, il faut que D soit $>= 0$.
Alors nous aurons: $x_1 = \frac{-b + \sqrt{D}}{2a}$ $x_2 = \frac{-b - \sqrt{D}}{2a}$

Imprimer x_1 et x_2

La <u>traduction</u> de cet algorithme dans un <u>programme</u> se trouve plus loin à l'exercice 16.

Si tu veux devenir un habile programmeur, je te conseille de toujours faire un algorithme. <u>Un problème</u> peut être résolu avec <u>plusieurs types d'algorithmes</u> et on peut dire que la découverte d'un nouvel algorithme est un acte créatif qui fait appel à l'intelligence, à l'expérience et aussi à l'intuition.

LES FONCTIONS (suite)

Dans ce chapitre nous continuons l'étude des **fonctions bibliothèques**. Le mot bibliothèque est employé parce que ces fonctions sont des programmes utilitaires, écrits à l'avance dans la mémoire du C64, et auxquels on accède simplement en signalant leur nom.

SQR

La fonction bibliothèque SQR permet de calculer la racine carrée d'un nombre qui forme l'argument de cette fonction. Ce nombre doit être >= 0.

Rappelle-toi qu'une fonction est toujours suivie de parenthèses entre lesquelles tu mets la valeur de l'argument. SQR est l'abréviation de SQuare Root (racine carrée).

EXERCICE 14 Imprime la racine carrée de 100 (en mode direct)

PRINT SQR(100) RETURN...L'écran affiche: 10.

Si tu écris:
PRINT SQR(-100) RETURN...Le C64 affiche: ILLEGAL QUANTITY ERROR

EXERCICE 15 Ecris un programme qui calcule la racine carrée des 20 premiers nombres.

```
10 PRINT"NOMBRES"TAB(10)"RACINE CARREE"
20 FOR X=1 TO 20
30 PRINT X TAB(15)SQR(X)
40 NEXT
```

NOTE Racine carrée d'un nombre équivaut à puissance .5 de ce nombre. Donc PRINT SQR(100) équivaut à PRINT 100↑.5

La fonction SQR n'est donc pas indispensable; elle est cependant utile et plus rapide que l'exponentiation.

EXERCICE 16 Ecris un programme qui calculera les racines d'une équation du second degré (voir l'algorithme page 132

```
10 PRINT"AX↑2+BX+C=0"
20 INPUT"VALEURS DE A  B ET C";A,B,C
30 IF A=0 AND B=0 AND C=0 THEN PRINT"INDETERMINE": END
40 IF A=0 AND B=0 THEN PRINT"IMPOSSIBLE": END
50 IF A=0 THEN PRINT"PREMIER DEGRE X=";-C/B: END
60 D=B↑2-4*A*C
70 IF D<0 THEN PRINT"PAS DE RACINES REELLES": END
80 X1=(-B+SQR(D))/(2*A)
90 X2=(-B-SQR(D))/(2*A)
100 PRINT"X1=";X1: PRINT"X2=";X2
110 END
```

Voici des **exemples de valeurs** attribuées à A, B, C et des résultats obtenus:

Entrées(valeurs de A,B,C)	**Exécution**(résultats)	**Equations**
0, 0, 3	Impossible	3 = 0
0, 0, 0	Indéterminé	0 = 0
0, 3, -6	Premier degré	3x-6=0 donc x=2
1, 2, 2	Pas de racines réelles	$x^2+2x+2=0$
1, -1, -6	X1 = 3 et X2 = -2	
1, -6, 9	X1 = 3.00006104 et X2 = 2.99993896	

Ici on sait que la réponse aurait du être une racine double: X1=3 et X2=3 ('erreurs d'arrondis', voir l'exercice 17).

EXERCICE 17 Puisque B↑2 correspond à B*B, amène le curseur sur la ligne 60 et change B↑2 par B*B.

Donne à A, B, C les mêmes valeurs que dans l'exemple précédent et tu auras X1=3 et X2=3.

Tu viens de découvrir que l'opération d'exponentiation peut introduire des erreurs qu'on appelle 'erreurs d'arrondis'.

EXERCICE 18 Comme dernier exercice avec SQR, écris un programme qui calcule la surface d'un triangle quelconque (formule de Haro).

```
10 PRINT"LONGUEUR DE CHACUN DES 3 COTES": PRINT"DU TRIANGLE"
20 INPUT A,B,C
30 P=(A+B+C)/2
40 C=(P*(P-A)*(P-B)*(P-C))
50 IF C<=0 THEN PRINT"CE N'EST PAS UN TRIANGLE!": END
60 S=SQR(C)
70 PRINT"SURFACE="S
```

END et STOP

END et STOP sont des instructions qui arrêtent le programme. Si tu regardes le listing de l'exercice 16 ci-dessus, tu retrouveras le mot END à 5 endroits. Note: END ne signifie pas seulement 'Fin', il veut dire **'retourner au niveau de commande directe'.**

La ligne 110 peut être effacée car le C64 arrête le programme automatiquement quand il a terminé l'exécution de toutes les lignes d'un programme.

END ou STOP servent à arrêter un programme en cours d'exécution. La seule différence entre les 2 instructions réside dans le message affiché sur l'écran. Avec END... READY. Avec STOP... BREAK IN...(no de ligne). Tu peux alors demander la valeur d'une variable en mode direct.

Nous verrons dans un deuxième volume de cette série qu'il existe une 2e différence: END ferme les fichiers ouverts et STOP ne les ferme pas.

END et STOP n'affectent pas les variables ni les pointeurs. Tu peux donc écrire CONT pour avoir la suite du programme. Mais si tu fais la moindre modification à ton programme, CONT ne peut plus être utilisée et la valeur des variables est perdue (remise à zéro).

Exemple: Le programme de l'exercice 16 recherche les racines réelles d'une équation du second degré. Si A, B, C valent respectivement 1, 2, 2, alors l'écran affiche: pas de racines réelles... READY (END de la ligne 70).

Fais le chargement (LOAD) de l'exercice 16 . Fais exécuter ton programme et inscris les données du paragraphe précédent... Appuie RETURN. Ecris maintenant CONT et appuie RETURN: L'écran affiche: ILLEGAL QUANTITY
ERROR IN 80

Ainsi tu peux savoir pourquoi il n'y a pas de racines relles...

Autre exemple: Si A, B, C valent 0, 3, -6 alors l'écran affiche: Premier degré X=2... READY. Ecris CONT et appuie RETURN: L'écran affiche:
DIVISION BY ZERO
ERROR IN 80

Ainsi tu peux savoir pourquoi il n'y a pas de racine du second degré.

INT( )

La fonction INT(X) nous donne la <u>partie entière</u> (anglais: INTeger) d'un nombre X qui forme l'argument de la fonction. Autrement dit un argument à virgule flottante est converti en un argument <u>entier</u>.

Cette fonction provoque <u>l'arrondi inférieur</u> de l'argument.

<u>EXERCICE 19</u> Ecris en mode direct:

```
PRINT INT(79.467)      RETURN...L'écran affichera 79     l'arrondi
PRINT INT(-15.248)     RETURN...L'écran affichera -16    inférieur
```

On se sert souvent de la fonction INT pour arrondir une valeur à un certain nombre de décimales. Par exemple 1 kilowattheure d'électricité coute 3,62 cents et 564 kWh couteront 3,62 X 564 = 20,4168 dollars. Il y a 4 décimales et tu veux en conserver seulement deux. De plus, il faudra arrondir à 20,42 dollars car la 3e décimale est >= 5.

<u>EXERCICE 20</u> Ecris en mode direct:



La valeur .5 permet d'arrondir le prix pour donner une précision à <u>1 cent près</u>. Dans notre exemple, la valeur .5 a permis d'arrondir le montant à la décimale supérieure. Si le montant avait été 20.4148 dollars la valeur .5 n'aurait rien changer et le montant aurait été arrondi à la décimale inférieure.

NOTE Si tu veux garder <u>une</u> décimale, tu remplaces les valeurs 100 par 10. Pour garder <u>trois</u> décimales tu mets 1000 au lieu de 100...etc.

EXERCICE 21 Tu prends une obligation du Québec de 5000 dollars à un taux d'intérêt annuel de 12% avec réinvestissement des intérêts chaque année pendant 10 ans. Quelle sera la valeur du chèque que le Gouvernement va t'adresser dans 10 ans?

```
1 REM SOIT C=LE CAPITAL, I=LE % D'INTERET ANNUEL
2 REM N=LE NOMBRE DE MOIS, V=MONTANT VERSE/MOIS
5 PRINT "♥"
```

```
10 INPUT "CAPITAL";C
15 INPUT "INTERET EN DECIMALES";I
20 INPUT "NOMBRE D'ANNEES";N
30 PRINT "ANNEE " TAB(10)"CAPITAL" TAB(22)"INTERETS"
40 FOR K=0TON
45 IF K=0 THEN 70
50 IN=C*I
60 C=C+IN
70 PRINT K TAB(10)INT(C*100+.5)/100 TAB(22)INT(IN*100+.5)/100
80 NEXT K
90 PRINT"CHEQUE: $";INT(C*100+.5)/100
```

Le programme est construit avec des variables afin que tu puisses entrer différentes valeurs pour le capital, les intérêts et le nombre d'années.

EXERCICE 22 Ecris la réponse de l'exercice 21 au moyen d'une seule instruction.

```
25 PRINT C*(1+I/100)↑N: STOP: REM FORMULE DES INTERETS COMPOSES
```

Une application pratique de la fonction INT( ) dans le **projet F-1 ci-dessous:**

PROJET F-1: Facture d'électricité d'Hydro-Québec.

Le problème: Faire un programme qui calcule le coût d'une facture d'électricité d'Hydro (1983). Ajouter la taxe provinciale et donner le montant total (compteur avec multiplicateur 10 et tarifs D1).

1- Exprime une solution du problème sous forme d'algorithme.
2- Traduis cet algorithme sous forme de programme (langage BASIC).
3- Fais imprimer les résultats sous les mots: COUT TAXE MONTANT

NOTES:
1- Avec ce projet F-1, tu ne pourras pas calculer la première facture de l'année car la période de consommation chevauche la fin de 1982 (anciens tarifs) et le début de 1983.
2- Les résultats seront parfois tronqués d'une ou deux décimales car le C64 n'imprime pas les derniers zéros des décimales...à moins que tu lui donnes des instructions précises à ce sujet.
3- A la suite du projet F-1, tu trouveras un projet F-2 qui **résoudra** les problèmes de chevauchement, d'arrondi, de tarifs G1, polyphasé ou non... etc.

Voici une solution du projet F-1:

<u>A- Algorithme</u>:

1. Les données: a) les constantes pour 1983 dans une instruction DATA

```
Soient: T1 = .245$:  la redevance d'abonnement quotidienne
        T2 = .029$:  le prix d'un kWh pour un maximum de
                     30 kWh/jrs à ce prix
        T3 =.0362$:  le prix de chaque kWh supplmentaire
        TX =  .09$:  la taxe prov
```

b) les données à entrer au clavier.

```
Soient:  L2   le relevé présent (lecture du compteur)
         L1   le relevé précédent
         J    le nombre de jours de consommation
```

2. Les calculs:

```
RA = J*T1  (redevance d'abonnement)

Si L2  L1, alors ajouter 10 000 à L2  (les 4 cadrans ont
                                        fait un tour)
Sinon KC = (L2-L1)*10  (multiplicateur 10..., nombre de
                         kWh consommés)
      KR = 30*J        (nombre de kWh à prix réduit)

Si KC   KR, alors PR = KC*T2  (prix à taux réduit)
                  CT = RA+PR  (cout total avec taux réduit)
                 ALLER à RESULTATS

Sinon PR = KR*T2  (prix réduit)
      PM = (KC-KR)*T3 (prix maximal)
      CT = RA+PR+PM  (cout total)
```

3. Les résultats:

```
TA = la valeur entière avec 2 décimales (arrondie à
 1 cent près) de (CT*TX)
CT = la valeur entière avec 2 décimales (arrondie
 à 1 cent près) de CT
MO = CT + TA

Imprimer   COUT + TAXE = MONTANT
Imprimer   CT; TA; MO
Fin
```

B- Le programme:

```
5 PRINT"
10 T1=.245: T2=.029: T3=.0362: TX=.09: REM TARIFS 1983
20 PRINT"RELEVE PRESENT":PRINT"RELEVE PRECEDENT":PRINT"NOMBRE DE JOURS
30 INPUT L2,L1,J
40 RA=J*T1: REM REDEVANCE ABONNEMENT
50 IF L2<L1 THEN L2=L2+10000: REM LES 4 CADRANS ONT FAIT 1 TOUR
60 KC=(L2-L1)*10: REM NOMBBRE DE KWH CONSOMMES
70 KR=30*J: REM KWH A PRIX REDUIT
80 IF KC<=KR THEN PR=KC*T2: CT=RA+PR: GOTO120
90 PR=KR*T2: REM PRIX REDUIT
100 PM=(KC-KR)*T3: REM PRIX MAX.
110 CT=RA+PR+PM: REM COUT TOTAL
119 REM***RESULTATS***
120 TA=INT(CT*TX*100+.5)/100: REM TAXE PROV. ARRONDIE
130 CT=INT(CT*100+.5)/100: REM COUT TOTAL ARRONDI
140 MO=CT+TA
150 PRINT" COUT  + TAXE =MONTANT"
160 PRINT CT;TA;MO
```

Avant de présenter le **projet F-2 et sa solution**, étudions les fonctions: LEN, STR$, et VAL ainsi que 2 instructions: GOSUB et RETURN que nous utiliserons dans le programme F-2.

LEN( ) **STR$( )** et **VAL( )**

La fonction **LEN** permet de connaître le nombre de caractères qu'il y a dans une chaîne. Cette fonction te donne la longueur (anglais: LENgth) d'une chaîne en te donnant **le nombre** de ses caractères.

EXERCICE 23) Ecris en mode direct:

```
PRINT LEN("Jean-Louis")
```

l'argument de la fonction LEN

L'écran affiche: 10 car la chaîne"Jean-Louis" a une longueur de 10 caractères.

Si tu veux savoir le nombre de caractères que possède une variable numérique, il faut d'abord utiliser la fonction STR$ dont le rôle est de représenter un nombre sous forme de chaîne de caractères.

EXERCICE 24 Ecris en mode programmé:

```
10 N=13245.86
20 N$=STR$(N)
30 PRINT LEN(N$)
```

L'écran affichera 9

Explications:
Ligne 10: met la valeur 13245,86 dans la variable numérique.

Ligne 20: la variable numérique N devient une variable chaîne N$ grâce à la fonction STR$.

Ligne 30: imprime le nombre de caractères que possède la chaîne N$. Etant donné que l'argument de la fonction LEN( ) est une chaîne, l'instruction no 20 devenait nécessaire pour transformer la variable numérique N en une variable chaîne N$ (ou variable alphanumérique).

QUESTION L'écran affiche 9 et je compte seulement 8 caractères dans la chaîne "13245.86"...Pourquoi?

Réponse: Le 9e caractère est celui de l'espace que le C64 réserve au début du nombre pour indiquer le signe du nombre (le signe + est sous-entendu).

NOTE Les lignes 20 et 30 ci-dessus peuvent être remplacées **par une seule** instruction:

20 PRINT LEN(STR$(N))

Application: Voir PROJET F-2, sous-programme 300 page 144.

La fonction VAL joue le rôle inverse de la fonction STR$, c'est-à-dire que VAL(N$) donnera la valeur numérique contenue dans la chaîne N$.

EXERCICE 25 Ecris en mode direct:

PRINT VAL(N$)

L'écran affiche 13245.86 c'est-à-dire la VALeur numérique représentée par la chaîne N$.

NOTE Les seuls caractères permis dans l'argument de VAL sont: les chiffres, l'espace, le signe +, le signe -, et le point... sinon VAL(N$)=0.

GOSUB et RETURN

L'instruction GOSUB 300 veut dire ALLER à SOUS-PROGRAMME (anglais: SUBroutine) ligne no 300...

Tu as étudié l'instruction GOTO... (ALLER à...) et tu l'as employée plusieurs fois. Regarde les lignes 110 et 120 page 94: L'instruction GOTO 30 permet d'ALLER à la ligne 30 et c'est à cette ligne 30 qu'est transféré le contrôle de l'exécution du programme.

L'instruction GOSUB transfère aussi le contrôle de l'exécution du programme à une section appelée sous-programme. Mais il y a une importante différence avec GOTO: l'instruction GOSUB garde en mémoire l'endroit où elle se trouve et aussitôt que l'ordinateur rencontre l'instruction RETURN, il RETOURNE à l'instruction qui suit immdiatement GOSUB.

EXERCICE 26 Ecris un programme qui utilise 8 fois la même boucle d'attente d'une seconde. Refais le même exercice en remplaçant les 8 boucles par un sous-programme.

1- Programme **sans** GOSUB (mauvaise programmation).

```
 10 PRINT"NOUVELLES EXPRESS": FORT=1TO1000:NEXT
 20 PRINT"APPARITION";: FORT=1TO1000:NEXT
 21 PRINT" D'UNE": FORT=1TO1000:NEXT
 22 PRINT"SOUCOUPE";: FORT=1TO1000:NEXT
 23 PRINT" VOLANTE": FORT=1TO1000:NEXT
 24 PRINT"LES";: FORT=1TO1000:NEXT
 25 PRINT" ENVAHISSEURS": FORT=1TO1000:NEXT
 26 PRINT"SONT LA!": FORT=1TO1000:NEXT
 40 END
```

2- Programme **avec** GOSUB (tu sauves plusieurs octets (mémoires).

```
10 PRINT"NOUVELLES EXPRESS": GOSUB100
20 PRINT"APPARITION";: GOSUB100
21 PRINT" D'UNE": GOSUB100
22 PRINT"SOUCOUPE";: GOSUB100
23 PRINT" VOLANTE": GOSUB100
24 PRINT"LES";: GOSUB100
25 PRINT" ENVAHISSEURS": GOSUB100
26 PRINT"SONT LA!": GOSUB100
40 END
100 FOR T=1TO1000:NEXT
110 RETURN
```

EXERCICE 27 Tu as probablement conservé sur cassette l'exercice 17 page 93. Introduis ce programme dans ton C64 et fais un sous-programme avec les lignes **75 à 100.**

```
10 PRINT"G A GAUCHE"
11 PRINT"H A DROITE
12 PRINT"APPUIE UNE TOUCHE
20 GETA$:IFA$=""THEN 20
25 PRINT"":POKE650,128
30 IFC<0THENC=0
35 IFC>33THENC=33
40 PRINTTAB(C)"   | "
45 PRINTTAB(C)"  ( )"
50 PRINTTAB(C)"/‾‾‾‾‾‾\"
55 PRINTTAB(C)"|______|"
60 PRINTTAB(C)"OOOOOO"
65 PRINT"";
70 GETB$:IFB$="G"THEN GOSUB75:C=C-1:GOTO 30
71 IFB$="H"THEN GOSUB75:C=C+1:GOTO 30
72 GOTO 70
75 PRINTTAB(C)"      "
80 PRINTTAB(C)"      "
85 PRINTTAB(C)"       "
90 PRINTTAB(C)"       "
95 PRINTTAB(C)"       "
100 PRINT"";:RETURN
```

Ne pas confondre
l'instruction RETURN
avec la touche RETURN

PROJET F-2 :

Calcul du montant d'une facture d'Hydro-Québec, tarifs D1 ou G1, multiplicateurs variés et courant monophasé ou polyphasé. Le calcul des coûts se fera à + ou - 1 cent.

Complète l'algorithme du PROJET F-1 de la page 137 pour arriver au programme F-2 ci-dessous. Il est possible, sans algorithme, d'organiser un programme qui "marche" après des essais multiples et plusieurs transformations... Mais ce programme deviendra un programme spaghetti difficile à lire et même l'auteur du programme ne s'y retrouvera plus s'il le relit quelques mois plus tard!

Ce programmme F-2 est formé de <u>6 sous-programmes</u> structurés autour d'un **programme principal.** Je te conseille de consacrer tout le temps cessaire pour bien comprendre le <u>déroulement séquentiel</u> de ce programme et l'enchaînement du **programme principal** avec **chacun des modules** formant les sous-programmes.

<u>PROGRAMME F-2</u>

> NOTE: En pratique, chaque client d'Hydro est inscrit dans un <u>fichier</u>(voir vol.2) et l'ordinateur a déjà en mémoire la réponse à la plupart des questions posées au début de ce programme.

```
5 PRINT"■"
10 INPUT"TARIFS ■D■ OU ■G■";TA$
15 IF TA$<>"D"AND TA$<>"G"THEN10
20 IF TA$="G"THEN GOSUB600
30 IF TA$="D"THEN GOSUB700
40 INPUT"■MULTIPLICATEUR..";MU
50 PRINT"■PREMIERE FACTURE DE L'ANNEE ■O■ OU ■N■"
55 INPUT PF$
60 IF PF$<>"O"AND PF$<>"N"THEN50
70 IF PF$="N"THEN JJ=0: GOTO 90
80 PRINT"■NOMBRE DE JOURS FACTURES AVANT 1 JANVIER"
85 INPUT JJ
90 PRINT"■■■RELEVE PRESENT?": PRINT"RELEVE PRECEDENT?"
95 PRINT"NOMBRE DE JOURS?■"
100 INPUTL2,L1,J
110 RA=J*T1: REM REDEVANCE ABONNEMENT
115 IF JJ<>0THEN RA=RA-JJ*(T1-T4): REM FACTURE JANVIER
120 IF L2<L1 THEN L2=10000+L2: REM LES 4 CADRANS ONT PASSE PAR 0000
130 KC=(L2-L1)*MU: REM NOMBRE KWH CONSOMMES
140 KR=30*J: REM NOMBRE KWH A PRIX REDUIT
150 IF KC<=KRTHEN GOSUB 500:GOSUB 200
160 IF KC>KRTHEN GOSUB 400:GOSUB 200
170 INPUT"■■AUTRE FACTURE ■O■ OU ■N■";AF$
180 IF AF$="O"THEN 5
190 END
199 REM--------------
200 REM IMPRESSION DES RESULTATS
205 PRINT"■■TARIFS "TA$"1 1983 +TX 9%"
210 PRINT"■■ COUT  + TAXE =MONTANT■"
```

```
220 X=CT:PRINTCT;:GOSUB300
230 X=TA:PRINTTA;:GOSUB300
240 X=MO:PRINTMO;:GOSUB300
245 PRINT
250 PRINT"██SI LE MONTANT DE TA FACTURE EST": PRINT"PLUS ELEVE"
260 PRINT"█TU ES PROBABLEMENT PENALISE!": PRINT"█TEL. A HYDRO"
270 RETURN
299 REM--------------
300 REM ZERO POUR AVOIR TOUJOURS 2 DECIMALES
310 LT=LEN(STR$(X))
320 LE=LEN(STR$(INT(X)))
330 IF LT-LE=2THENPRINT"█0 ";
340 IF LT-LE=0THENPRINT"█.00 ";
350 RETURN
399 REM -------------
400 REM  CALCULS SELON TARIFS MAXIMUMS
410 PR=KR*T2: REM PRIX REDUITS
415 IF JJ<>0THEN PR=PR-(KR*JJ/J)*(T2-T5): REM FACT.DEBUT ANNEE
420 PM=(KC-KR)*T3: REM PRIX MAX.
425 IF JJ<>0THEN PM=PM-(KC-KR)*(JJ/J*(T3-T6)): REM FACT.DEBUT ANNEE
430 TA=INT(((RA+PR+PM)*TX+.0099)*100)/100: REM TAXE ARRONDIE A 2 DEC.
440 CT=INT((RA+PR+PM+.0025)*100)/100: REM COUT TOTAL ARRONDI A 2 DEC.
450 MO=CT+TA: REM MONTANT TOTAL
460 RETURN
499 REM--------------
500 REM  CALCUL SELON TARIFS REDUITS
510 PR=KC*T2: REM PRIX REDUITS
515 IF JJ<>0THEN PR=PR-KC*JJ/J*(T2-T5): REM FACTURE DEBUT ANNEE
520 TA=INT(((RA+PR)*TX+.0099)*100)/100: REM TAXE ARRONDIE A 2 DEC.
530 CT=INT((RA+PR+.0025)*100)/100: REM COUT TOTAL ARRONDI A 2 DEC.
540 MO=CT+TA: REM MONTANT TOTAL
550 RETURN
599 REM--------------
600 REM TARIFS G1
610 T1=.245:T2=.0483:T3=.028:TX=.09: REM 1983
620 T4=.23:T5=.045:T6=.026: REM 1982
630 INPUT"██POLYPHASE █O█ OU █N█";PO$
640 IF PO$<>"O"AND PO$<>"N"THEN630
650 IF PO$="O"THEN T1=.735
660 RETURN
699 REM -------------
700 REM  TARIFS D1
710 T1=.245:T2=.029:T3=.0362: REM ANNEE 1983
720 T4=.23:T5=.027:T6=.0337:TX=.09: REM 1982
730 RETURN
```

Explication des lignes 300 à 350: sous-programme des 2 décimales

Il faut d'abord noter que si tu écris:

PRINT 23.00   l'écran affiche **23** et
PRINT 23.50   l'écran affiche 23.5.

Donc le C64 élimine automatiquement les caractères qui n'affectent pas le résultat obtenu: soient .00 et 0.

Il est cependant nécessaire d'indiquer ces zéros sur une facture et c'est le but du sous-programme des lignes 310 à 350.

Ligne 310: LT = LEN(STR$(X))

- STR$: **Fonction** qui transforme une valeur numérique en une chaîne de caractères
- LEN : **Fonction** qui donne le <u>nombre</u> de caractères qu'il y a dans la chaîne X.
- LT : **Variable** dans laquelle s'inscrira le <u>nombre</u> de caractères que contient la Longueur Totale de la chaîne.

Ligne 320: LE = LEN(STR$(INT(X)))

- INT : **Fonction** qui donne la <u>partie</u> entière d'un nombre.
- STR$: **Fonction** qui transforme la partie entière du nombre en une chaîne.
- LE : **Variable** dans laquelle s'inscrira le nombre de caractères que contient la partie <u>E</u>ntière de la chaîne.

Ligne 330: IF LT-LE = 2 THEN PRINT" 0 "

- espace
- curseur à gauche (No 4 page 35)

Exemple: 128.5 dollars. Dans ce cas:
LT=6 (5 caractères **plus 1** pour le signe)
LE=4 (3 caractères **plus 1** pour le signe)
Donc l'instruction 330 ajoutera un zéro au montant, soit 128.50 dollars.

Il faut noter aussi que **les tarifs** pour les années 1983 et 1982 sont **placés à la fin**, dans un sous-programme, pour faciliter les changements de tarifs au début d'une nouvelle année.

RND( )

La fonction RND (anglais: RaNDom ⇒ hasard) permet d'obtenir des nombres aléatoires compris entre 0 et 1 (c'est-à-dire 0 et 0.99999...). L'argument de RND peut prendre différentes valeurs, mais le plus souvent on utilise RND(1).

<u>EXERCICE 28</u> Programme qui donne 10 nombres 'tirés au hasard'

```
10 FOR I=1 TO 10
20 PRINT RND(1)
30 NEXT
```

Exécution du programme:
L'écran affiche 10 nombres 'tirés au hasard': .185564016
.0468986348

NOTE Au départ on aura toujours ces résultats si **RND** a un argument supérieur à zéro.

.827743801
.554749226
.897233831
.572916248
.838893164
.931229627
.188382009
.97293994

Il y a beaucoup d'applications intéressantes qui font appel à la génération de nombres aléatoires.

Par exemple, la fonction RND est souvent utilisée...1- pour simuler des **jeux de hasard**, 2- pour fournir un nombre **entier** compris entre **deux valeurs**, 3- dans des applications statistiques...etc.

Premier exemple: Lancement d'un dé.

Pour simuler le lancement d'un dé, l'ordinateur devra fournir une valeur aléatoire et entière comprise entre 1 et 6.

EXERCICE 29 Ecris en mode direct:

```
PRINT INT(RND(1)*6+1)
```

Explications:
a) Le nombre aléatoire obtenu par RND(1) est compris entre 0 et 0.9999...
Lorsqu'on le multiplie par 6, le nombre est compris entre 0 et 5.9999...
b) Après avoir additionné 1 à cette valeur, notre nombre est maintenant
compris entre 0 et 6.99999...
c) Enfin, la fonction INT nous donne la valeur entière du nombre. Cette
valeur sera un chiffre entier compris entre 1 et 6.

Deuxième exemple: Lancement de deux dés.

Pour que l'ordinateur puisse simuler le lancement de deux dés, il faudra écrire 2 fois la formule ci-dessus et additionner les deux chiffres obtenus au hasard et dont la somme minimale sera 2 et maximale 12.

EXERCICE 30 Ecris en mode direct:

```
PRINT INT(RND(1)*6+1) + INT(RND(1)*6+1)
```

EXERCICE 31 Ecris une fois + 2 au lieu d'écrire deux fois + 1 pour sauver deux octets (bytes) et la formule du lancement de 2 dés deviendra:

```
PRINT INT(RND(1)*6) + INT(RND(1)*6) + 2
```

NOTE La fonction RND génère des nombres qui ne sont pas réellement aléatoires puisque ces nombres proviennent d'un processus de calcul déterminé. Etant donné que ces nombres possèdent les mêmes propriétés mathématiques que ceux qui sont réellement aléatoires, on dit parfois que ce sont des valeurs pseudo-aléatoires.

Troisième exemple: Dans les exemples précédents, l'argument X de la fonction RND(X) était plus grand que 0 (>0) et on obtenait, au départ, la même suite de valeurs aléatoires (utile pendant la mise au point d'un programme... mais qui offre des inconvénients après...)

Si X = 0 on obtient une suite différente de valeurs aléatoires à chaque départ car dans le cas où X = 0 les nombres aléatoires sont générés par une horloge incorporée au C64.

EXERCICE 32 Tu lances dix fois un dé. Ecris un petit programme qui fait imprimer les 10 résultats.

```
10 FOR I=1 TO 10
20 PRINT INT(RND(0)*6+1)
30 NEXT
```

Résultats: 1 6 6 4 5 4 5 5 3. Dans chaque cas, tu obtiens un chiffre entier compris entre 1 et 6. Note tes résultats puis éteins et rallume le C64. Ecris de nouveau les instructions précédentes et tu obtiendras une nouvelle suite de valeurs aléatoires.

QUESTION Comment faire pour qu'aucun nombre ne se répète dans une suite de valeurs 'tirées au hasard'?

Réponse: Ce problème sera résolu dans l'exercice 36.

EXERCICE 33 Ce programme permet de rechercher un nombre aléatoire fourni par l'ordinateur (jeu).

```
5 C=0
10 PRINT"TU DOIS TROUVER UN NOMBRE ENTRE 0 ET...?"
15 INPUT"NOMBRE MAXIMUM";M
20 NA=INT(RND(0)*99+1): REM GENERATION DE NOMBRES ALEATOIRES
30 INPUT"QUEL NOMBRE";NC
35 C=C+1
40 IF NC>NA THEN PRINT"TROP HAUT"
50 IF NC<NA THEN PRINT"TROP BAS"
60 IF NC<>NA THEN 30
70 PRINT"FELICITATION!  ";NC
75 PRINT"TU AS TROUVE APRES":PRINTC"ESSAIS"
80 INPUT"1 AUTRE FOIS O OU N";P$
90 IF P$="O"THEN 5
100 END
```

LOI des GRANDS NOMBRES:

Si tu lances une pièce de monnaie en l'air, tu as autant de chances d'obtenir le côté face que le côté pile. Tu peux avoir le même côté plusieurs fois de suite mais si tu lances la pièce un **grand nombre de fois**, les probabilités nous enseignent que les résultats vont osciller autour de 50% pile et 50% face.

Il serait long de le vérifier "sur la main"...Ton ordinateur peut te donner le résultat de cent essais en moins de deux secondes!

EXERCICE 34 Ecris un programme qui joue à PILE ou FACE et fais imprimer les résultats c'est-à-dire le nombre de fois PILE et le nombre de fois FACE.

```
5 PRINTCHR$(147): REM EFFACER L'ECRAN
10 P=0: F=0
20 INPUT"NOMBRE DE LANCEMENTS";N
30 FOR I=1 TO N
40 CA=INT(RND(1)*2): REM NOMBRE ALEATOIRE SERA 1 OU 0
50 IF CA=1 THEN F=F+1
60 IF CA=0 THEN P=P+1
70 NEXT I
80 PRINT:PRINT F"FACES",P"PILES"
90 GOTO10
```

> La loi des grands nombres: "... plus il y a d'essais, plus on se rapproche de la probabilité d'arrivée..." Ici cette probabilité d'arrivée est déterminée à 50% pile et 50% face.

Résultats d'une expérience:

100 essais ont donné	45,5% face	54,5% pile
1000 essais ont donné	46,8% face	53,2% pile
10000 essais ont donné	50,46% face	49,54% pile
100000 essais ont donné	50,16% face	49,84% pile ...Ici

il faut au moins une demi-heure à l'ordinateur... Tu as le temps d'aller dîner!

EXERCICE 35 Ecris un petit programme qui fait sortir au hasard un nombre de 4 chiffres et dont chacun des 4 chiffres sera 'tiré' au hasard. Fais imprimer ces 4 chiffres en caractères inversés sur une même ligne (LOTO)

```
10 FOR I=1 TO 4
20 C=INT(RND(0)*10)
30 PRINT CHR$(18)C;
40 NEXT I
```

EXERCICE 36 Réponse à ta question de la page précédente. Le programme ci-dessous permet de sortir au hasard une liste de nombres compris entre 10 et 99 en s'assurant qu'on n'obtient pas 2 fois le même nombre Sois patient car il faut près de 2 minutes pour obtenir les 90 nombres différents!

```
5 PRINT"♥": DIM R(90): PRINT CHR$(5)
10 FOR I=1 TO 90
20 R(I)=INT(RND(0)*90+10)
25 K=R(I)
30 FOR J=I TO 1 STEP-1
40 IF K=R(J-1) THEN 20
50 NEXT J
60 PRINT "■"R(I);
65 NEXT I
```

L'instruction DIM à la ligne 5 et la <u>variable indicée R(I)</u> à la ligne 20 sont des notions nouvelles et <u>importantes</u>. Prends le temps nécessaire pour bien comprendre les <u>explications de la page suivante.</u>

<u>**EXERCICE 37**</u> Après avoir écrit LIST et appuyé RETURN, <u>enlève</u> l'instruction 65 de l'exercice 36 et ajoute les instructions 64 à 95 pour obtenir une meilleure présentation à l'écran.

```
64 REM*INSTRUCTIONS  POUR AFFICHAGE ECRAN*
65 C=C+1
70 IF C=10 THEN PRINT"■"
75 IF C=20 THEN PRINT CHR$(5): C=0
80 NEXT I
85 PRINT:PRINT"NOMBRES 10 A 99 'TIRE' AU HASARD...";
90 PRINT"ILS SONT TOUS DIFFERENTS";
```

SAUVE ce programme(lignes 5 à 95) sur cassette ou sur disque. Il te sera utile dans une prochaine expérience.

Dans l'exercice 36 que tu viens de 'sauver' (SAVE), il fallait s'assurer que les nombres obtenus au hasard <u>ne se répètent pas</u>.

Nous avons utilisé le moyen très simple qui consiste à inscrire dans un tableau à une dimension (ou liste) les nombres obtenus au hasard et à refuser les nombres dont les valeurs existent déjà dans la liste.

Donc chaque nombre obtenu au hasard (ligne 20) est comparé à chacun des nombres inscrits dans la liste (lignes 30 à 50). Si un nombre identique est déjà dans cette liste, (ligne 40), le C64 redemande un nombre différent au hasard et lorsqu'il obtient ce nombre différent des autres, celui-ci est retenu dans la liste R(I) et affiché à l'écran (ligne 60).

DIM

Le langage BASIC ne possède pratiquement que 2 instructions de <u>déclaration</u>: DATA et DIM.

DIM vient du mot DIMension. Son rôle est de déclarer la "<u>dimension</u> d'une <u>liste</u> ou d'un <u>tableau</u>" c'est-à-dire le nombre maximum d'éléments que contiendra ce tableau.

Le numéro de l'élément le plus bas est <u>toujours zéro</u> et le numéro de l'élément le plus élevé est le nombre déclaré dans DIM.

Dans un programme, l'instruction DIM doit être **exécutée avant toute opération sur la variable de DIM.** Cependant, cette instruction **<u>DIM ne doit être exécutée qu'une fois</u>** pendant le déroulement du programme.

NOTE DIM est obligatoire <u>seulement</u> si le nombre d'éléments est supérieur à 11 (0 à 10). Si le nombre d'éléments est égal ou inférieur à 11, le C64 va DIMensionner <u>automatiquement</u> le tableau.

On voit ici l'avantage et aussi la nécessité de **conserver des valeurs dans un tableau** afin de pouvoir les **comparer**, les **trier**, ... etc. Examinons ça de plus près!



Photo du circuit électrique : Didacticiel pages 154 à 156.

TABLEAUX à une DIMENSION ou LISTE (en mathématique: vecteur)

Dans l'exercice 36 ci-dessus nous avons employé pour la première fois une **variable indicée** R(I) avec une instruction DIM R(90).

Jusqu'à maintenant, nous utilisions des <u>variables simples</u> qui représentaient <u>une seule valeur</u>, donc <u>une seule position mémoire</u>.

Désormais tu pourras, **avec une seule variable indicée, représenter toute une liste de valeurs,** donc une succession de mémoires dont l'ensemble formera <u>une zone</u>.

Supposons que le programme de l'exercice 36 t'a fourni les **valeurs** aléatoires suivantes: (**éléments** du tableau)

<u>Tableau monodimensionnel</u> (ou liste)

R(0)	
R(1)	72
R(2)	47
R(3)	21
R(4)	49
R(5)	96
⋮	68
⋮	⋮
⋮	35
R(86)	58
R(87)	13
R(88)	24
R(89)	60
R(90)	83

Dans notre programme, ce tableau **à une** dimension **est représenté par la variable indicée R(I).**

Il est important de noter que:
1- **R est le nom** de ce tableau.
2- **(I) est l'indice** qui peut prendre ici les valeurs 0 à 90 pour indiquer l'emplacement de l'élément dans le tableau. On peut prendre <u>une autre lettre</u> comme indicateur d'indice, soient J, K, L,...etc
3- L'interpréteur du c64 commence toujours un tableau **par l'indice 0.**
4- Les nombres aléatoires forment ici les **éléments** du tableau. **Ces éléments s'appellent les variables indicées.**
5- Il faut dimensionner le tableau par une instruction DIM suivie du numéro de l'élément le plus élevé: DIM(90). On peut mettre une variable comme indicateur de l'élément le plus élevé.
Exemple: DIM R(N)

NOTE Dans ton programme de l'exercice 36, tu n'as pas mis d'élément dans la variable C(0) afin de faire coincider l'<u>élément un</u> de la zone avec <u>la valeur un</u> de la variable de contrôle I.

<u>EXERCICE 38</u> 1- Fais le chargement (LOAD) du programme de l'exercice 36. 2- Ecris RUN et tape RETURN. 3- Ecris sur un bout de papier quelques-uns des résultats affichés sur l'écran, par exemple les 5 premiers et les 5 derniers nombres...

Tu as maintenant, dans une zone mémoire, une liste de 90 nombres aléatoires (10 à 99).

Le nom de cette liste est R et chacun des éléments de la liste est représenté par une variable indicée R(1), R(2), R(3)... R(86),... etc.

<u>EXERCICE 39</u> **Vérifie, en mode direct, si tu as bien en** mémoire les valeurs notées sur ton bout de papier:

```
PRINT R(1)        RETURN...     PRINT R(2)        RETURN
```

<u>EXERCICE 40</u> Utilise une boucle FOR... NEXT et écris en mode direct:

```
FOR J=86 TO 90:PRINT R(J);:NEXT J      RETURN...
```

Tu obtiens les 5 derniers nombres du tableau.

Conserve ton tableau dans la mémoire du C64 car tu en auras besoin pour l'exercice 41.

<u>La SOMME et la MOYENNE des éléments de ta liste</u>:

Pour obtenir la <u>somme</u> des 90 nombres de ta liste, tu utilises une boucle FOR... NEXT qui fera 90 fois l'opération suivante: S=S + R(I). A la fin, la variable S contiendra la <u>somme</u> totale des 90 nombres.

Pour avoir la <u>moyenne</u> il faut diviser la somme par 90: M=S/90

<u>EXERCICE 41</u> Ecris en mode direct:

```
FOR I=1 TO 90: S=S+R(I): NEXT I: M=S/90: PRINT S,M
```

Tu obtiens: S=4905 et M=54,5

<u>EXERCICE 42</u> Fais entrer ces 3 instructions à l'intérieur du programme des exercices 36+37:

```
78 S=S+R(I)
85 M=S/90
93 PRINT:PRINT"S=" S,"M=" M
```

Les variables S, M, R(I),... sont automatiquement remises à zéro quand tu écris RUN et tapes RETURN

<u>Une LISTE de CHAINES de CARACTERES</u>

Il est aussi possible d'obtenir un tableau de chaînes de caractères. Il suffit alors d'utiliser **un identificateur qui se termine par le caractère <u>$</u>.**

EXERCICE 43 Ecris un programme qui utilise l'instruction INPUT pour entrer les noms et prénoms de cinq personnes et une variable indicée pour mettre ces noms (chaînes de caractères) dans un tableau

```
10 FOR I=1 TO 5
20 PRINT"NP$("I")"
30 INPUT"NOM PRENOM";NP$(I)
40 NEXT
```

Les cinq noms et prénoms forment maintenant une liste qui se nomme NP$ et chacun des noms et prénoms correspond à une variable indicée: NP$(1), NP$(2),... etc.

Dans un tableau, **l'élément varie selon la valeur de l'indice. Il est donc avantageux d'afficher** le nom de ce tableau (variable chaîne) et la valeur de l'indice... C'est le rôle que joue la ligne 20 de l'exercice 43.

EXERCICE 44 Utilise le mode direct pour faire imprimer les cinq noms et prénoms que tu viens d'entrer dans la mémoire du C64.

```
FOR K=1 TO 5:PRINT NP$(K):NEXT          RETURN
```

EXERCICE 45 Introduis (INST) le numéro de ligne 50 devant FOR dans l'exercice 44 (n'oublie pas de taper RETURN).

Ton programme peut maintenant afficher le tableau sur l'écran. Ecris LIST et tape RETURN pour vérifier si la ligne 50 est bien dans ton programme. Ensuite fais exécuter ton programme:(RUN RETURN). Les 5 noms et prénoms apparaissent sur l'écran.

Note importante Si tu veux mettre les noms de tous les élèves d'une classe dans un tableau, par exemple 25 élèves, il faudra ajouter l'instruction: **DIM NP$(25)**

Si tu veux donner plus de souplesse à ton programme, remplace 25 par une variable numérique, par exemple N: **DIM NP$(N)**

Il faudra alors donner la valeur de N avant l'instruction DIM. Ne pas oublier de remplacer 1 TO 5 par 1 TO N:

```
2 INPUT"NOMBRE D'ELEVES";N
10 FOR I=1 TO N
```

EXERCICE 46 En tenant compte de la note importante signalée ci-dessus, écris un programme qui met les noms et prénoms des élèves d'une classe dans un tableau.

```
2 INPUT"NOMBRE D'ELEVES";N
5 DIM NP$(N)
10 FOR I=1 TO N
20 PRINT"NP$("I")"
30 INPUT"NOM PRENOM";NP$(I)
40 NEXT
50 FOR K=1 TO N: PRINT NP$(K): NEXT
```

Les tableaux à 2 dimensions seront étudiés au chap. 5

ON...GOTO et ON...GOSUB

ON... GOTO et ON... GOSUB sont des instructions **d'aiguillages multiples**. Les aiguillages multiples peuvent être faits avec plusieurs IF... THEN, cependant on peut éviter la répétition d'une suite de IF... THEN par une seule instruction ON... GOSUB ou (GOTO) qui peut être considérée comme une variante de IF... THEN. Exemple: ON X GOTO 400, 500, 800, 600

Cette instruction veut dire: Si la variable **X vaut 1**, le programme sera orienté vers la ligne 400, c'est-à-dire le **premier numéro de ligne** noté après GOTO. Si X vaut 3 l'aiguillage se fera vers 800... etc.

Exemple pratique:

Dans le didacticiel des pages suivantes, nous avons un MENU qui offre 6 choix (1-6). La variable qui représente le MENU est M.

```
IF M = 1 THEN GOSUB 50
IF M = 2 THEN GOSUB 150   <---- Ces 6 instructions sont remplacées
IF M = 3 THEN GOSUB 250         par une seule à la ligne 25:
IF M = 4 THEN GOSUB 40
IF M = 5 THEN GOSUB 40          25 ON M GOSUB 50,150,250,40,40,30
IF M = 6 THEN GOSUB 30
```

DIDACTICIEL

Le programme qui occupe les deux pages suivantes est un exemple de "didacticiel". Il a été construit uniquement avec les notions "BASIC" que tu as apprises dans les pages précédentes.

Un **didacticiel** est **un programme informatisé** qui a pour **but d'instruire** de façon agréable la personne qui échange avec l'ordinateur pendant l'exécution de ce programme.

Ce volume 1 te donne la première partie du didacticiel. Tu trouveras la deuxième partie dans le volume 2: (le calcul des résistances en parallèle, le déplacement des électrons, la notation...etc).

Pour devenir habile dans la programmation, il faut créer un grand nombre de petits programmes (modules) plutôt que un ou deux longs programmes...Tu pourras ensuite juxtaposer plusieurs modules pour obtenir un programme plus élaboré.

Fais le chargement (LOAD) **de ton programme E-1** (page 124) **et utilise-le comme sous-programme dans ce didacticiel:**

DIDACTICIEL : Exercices sur la 'LOI d'OHM'.

```
1 PRINT" DIDACTICIEL"
2 PRINT SPC(9)"LOI D'OHM (EXERCICES)"
3 PRINT SPC(13)"PAR L.PICARD"
4 PRINT SPC(11)"APPUIE UNE TOUCHE"
5 GETA$:IFA$=""THEN5
6 PRINT""SPC(18)"MENU"
7 PRINT"1- UNE RESISTANCE (RHEOSTAT)"
8 PRINT"2- 2 RESISTANCES EN SERIE"
9 PRINT"3- UNE RESISTANCE EN SERIE ET DEUX EN" SPC(6)"PARALLELE"
10 PRINT"4-5-6- BYE BYE"
12 E=1024: CO=55296: S=54272: POKE S+24,15
15 INPUT"TAPE UN CHIFFRE(1-6)";M:IFM<1ORM>6THEN15
20 ON M GOTO50,150,250,40,40,30
30 PRINT"AUREVOIR
40 PRINT"SUITE DANS VOLUME 2...": END
50 J$="":A$=" |":B$=" |":R$="   |":K$="":GOSUB500
52 PRINT: PRINT"FERME LE CIRCUIT: I
55 GET C$: IF C$<>"I" THEN55
59 GOSUB 400: PRINT"FAIS VARIER + OU - RH ET APPUIE Q";
61 PRINT D$ TAB(C): POKE 650,128
65 GET H$: IF A=1 THEN 69
66 IF H$="+" THEN GOSUB 350: C=C+1: GOSUB 450: GOSUB 300
67 IF A=7 THEN GOSUB 550: GOTO 5
69 IF H$="-" THEN GOSUB 350: C=C-1: GOSUB 450: GOSUB 300
70 IF TI/60-H=>1 AND A>5 THEN POKE S+4,0: GOSUB 550: GOTO 5
72 IF H$="Q" THEN 80
75 GOTO 65
80 PRINT"QUELLE EST LA VALEUR EN OHMS
82 PRINT"DE LA RESISTANCE VARIABLE RH (RHEOSTAT)?"
85 INPUT R: U=12: RH=U/A: IFR=RH THEN 98
88 PRINT"JE REGRETTE
90 PRINT"DES EXPLICATIONS! TAPE O OU N
92 GET E$: IF E$="O" AND Z=1 THEN 105
93 IF E$="O" THEN 100
94 IF E$="N" THEN GOTO 5
96 GOTO 92
98 PRINT:PRINT RH"OHMS BRAVO!":GOTO 90
100 PRINT"POUR OBTENIR LA VALEUR D'UNE RESISTANCE R
101 PRINT"IL FAUT DIVISER LA TENSION (OU VOLTAGE) U
102 PRINT:PRINT"AUX BORNES DE CETTE RESISTANCE PAR"
103 PRINT:PRINT"LE COURANT I QUI LA TRAVERSE"
104 PRINT:PRINT"LOI D'OHM: U/I=R"
105 PRINT"" U "VOLTS": PRINT"---------- =" RH "OHM(S)"
106 PRINT A"AMPERE(S)"
110 PRINT"TAPE UNE TOUCHE
115 GET A$: IF A$="" THEN 115
117 IF Z=1 THEN Z=0: RETURN
120 GOTO 5
150 J$="":A$="/":B$="\":R$="R1\":K$="": GOSUB 500
155 D=INT(RND(0)*13)
156 FOR K=0 TO 12 STEP 2: IF D=K THEN 165
160 NEXT: IF D<>K THEN 155
165 PRINT D$;: IF D=12 THEN A=.5: U=6
170 IF D=10 THEN A=.1: U=1
```

Ne pas confondre le curseur du rhéostat et le curseur qui clignote sur l'écran. Le curseur du rhéostat se déplace sur la résistance et la fait + ou - varier.

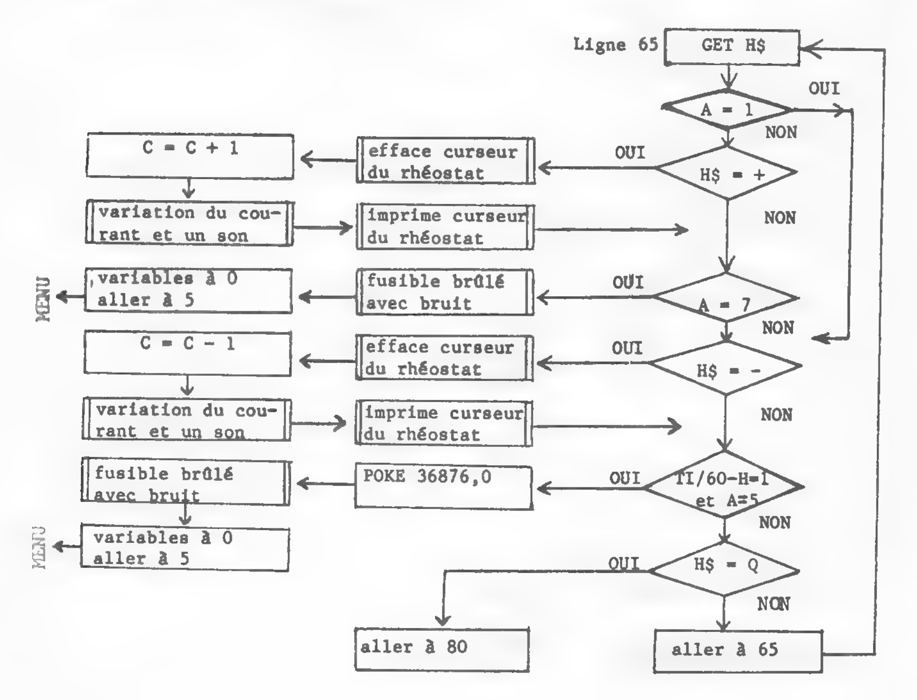
```
175 IF D=8 THEN A=1: U=8
177 IF D=6 THEN A=.5: U=3
179 IF D=4 THEN A=2: U=8
181 IF D=2 THEN A=3: U=6
185 IF D=0 THEN A=5: U=0
190 FOR I=1 TO 6: IF C=15+D/2 THEN 210
195 GOSUB 330: GOSUB 350: C=C-1: GOSUB460: GOSUB300
200 NEXTI
210 GOSUB 330: GOSUB 410: PRINT"    "TAB(4) A TAB(15) U " "
215 INPUT"        RESISTANCE R1= ";R
220 RT=12/A: RH=U/A: R1=RT-RH: IFR=R1 THEN PRINT"BRAVO R1="R"OHMS";: GOTO 2
225 PRINT"JE REGRETTE!": FOR T=1 TO 1500: NEXT
226 PRINT"       "TAB(24 )"L'ACCUMULATEUR":PRINTTAB(24)"DONNE 12 VOLTS "
227 PRINT TAB(24) U"V PERDU(S)": PRINTTAB(24)"DANS RH "
228 PRINT TAB(24)"IL RESTE " 12-U " V":PRINTTAB(24)"POUR R1 "
229 PRINT TAB(24)" " A " AMPERE(S)": PRINTTAB(24)"TRAVERSE(NT)R1 "
230 PRINT TAB(27)"LOI D'OHM:":PRINTTAB(27)"R1=U/I "
231 PRINT TAB(27)"R1=" 12-U"/"A:PRINTTAB(27)" R1="R1"OHMS  "
235 PRINT TAB(19)"   TAPE UNE TOUCHE";
238 GET E$: IF E$="" THEN 238
240 GOTO 5
250 J$="_":A$=" / \ ":B$=" \ / ":R$=" R1\ /R2 ":K$="‾":GOSUB500
260 PRINT"  SUITE DANS VOLUME 2... ": END
300 PRINT TAB(C)" .": PRINT TAB(C)CHR$(180): PRINT"  ";: RETURN
330 FOR T=1 TO 1000: NEXT: RETURN
350 G$="‾": IF H$="+" THEN G$=" "
360 PRINT TAB(C)G$: PRINT TAB(C)" ": PRINT"  ";: RETURN
400 A=1: POKE E+565,177: POKE E+575,177: POKE E+576,178
410 POKE S+5,0: POKE S+6,240
415 POKE S+1,25: POKE S,177: POKE E+129,32: POKE E+169,121
417 POKE S+4,129
420 POKE CO+169,0: POKE S+4,128: RETURN
450 A=22-C: K=176+A: POKE E+565,K
460 POKE S+1,28: POKE S,214: POKE S+4,17: IF A>5 THEN H=TI/60: RETURN
470 POKE S+4,16: POKE S+1,0: POKE S,0: RETURN
500 PRINT" ": POKE 53280,1: POKE 53281,1
501 PRINT" FUSIBLE          ACCU"
502 PRINT"            I     MULATEUR"
503 PRINT"    5A      _      +     -"
504 PRINT"____/\_____  ______.      ._
505 PRINT"|                     12        |
506 PRINT"|                               |
507 PRINT"|                     VOLTS     |"J$
508 PRINT"|                               "A$
509 PRINT"|                            "B$
510 PRINT"|                            "A$
511 PRINT"| AMPEREMETRE                "B$
512 PRINT"|                  VOLTMETRE"A$
513 PRINT"|    /‾‾‾\         ‾‾‾‾      "B$
514 PRINT"|       0  A|      |0  V|      "A$
515 PRINT"|    |     |       |____|   "R$
516 PRINT"|    L_____J                    |"K$
517 PRINT"|    | |   | |              L___ |
518 PRINT"‾‾‾‾‾‾    ‾‾‾‾‾‾‾‾‾‾‾‾‾           ‾‾"
519 PRINT"                  |         | RH"
520 PRINT"          RHEOSTAT XXXXXX  "
525 D$="                        ": C=21: RETURN
```

```
550 POKE E+163,73: POKE E+164,74: POKE S+1,25: POKE S,177
551 POKE S+4,129
555 FOR T=1 TO 200: NEXT: POKE S+4,128
560 POKE E+565,48: POKE E+575,48: POKE E+576,32
565 PRINT D$"■■■■■ PLUS DE 5■ AMPERES!!! APPUIE ■↑■ ET ■F■";
570 IF A=7 THEN PRINT"■■■■■■■■■■■■COURT-CIRCUIT!
575 GET I$: IF I$="↑" THEN POKE E+169,32: POKE E+129,121
580 IF I$="F" THEN POKE E+163,105: POKE E+164,95
585 IF PEEK(E+164)=95 AND PEEK(E+169)=121 THEN 550
590 IF PEEK(E+164)=95 AND PEEK(E+169)=32 THEN 599
595 GOTO 575
599 GOSUB 330:PRINT"■" SPC(219)"OK" SPC(195)"CONTINUE"
600 PRINT SPC(172)"TAPE  UNE TOUCHE": GOSUB 330: RETURN
```

Ce "didacticiel" résume bien les notions "BASIC" que tu as apprises dans les 4 premiers chapitres et tu possèdes tous les éléments nécessaires pour interpréter chacune des lignes de ce programme.

Pour participer à ton effort de compréhension, nous ferons l'organigramme des lignes 65 à 75.

<u>ORGANIGRAMME des LIGNES 65 à 75</u>:



CHAPITRE V

ACTIVITES DIRIGEES Série no 5

BUT

Cette dernière série d'activités te permettra d'apprendre:

1- Une interprétation du listing: "Le Fleur de Lis" fourni par l'imprimante VIC-1525.

2- Les entrées clavier. La fonction PEEK(197) et le tableau page 165 des numéros des touches du clavier.

3- Les tableaux à deux dimensions.

4- Les variables entières.

5- Les instructions DEF FN et WAIT.

6- Les fonctions trigonométriques: SIN, COS, TAN et ATN.

7- Le tracé de la courbe représentative d'une fonction.

8- Les fonctions d'extraction: RIGHT$, LEFT$, et MID$.

9- Le mode texte.

UN SOUS-PROGRAMME : **Le Fleur de Lis.**

Il arrive souvent qu'un programme débute par l'affichage d'un dessin sur l'écran. Dans les programmes de ce chapitre 5 nous avons choisi le Fleur de Lis. Je t'encourage à faire cet exercice et à le conserver sur cassette en vue de l'utiliser comme sous-programme.

Tu peux travailler selon la méthode des PRINT, comme dans les dessins précédents, et améliorer celui qui t'est suggéré ci-dessous. Ce sera plus joli en haute résolution (vol 2).

Listing

Sur l'écran, après RUN et RETURN.

```
100 PRINT"
101 PRINT"
102 PRINT"
103 PRINT"
104 PRINT"
105 PRINT"
106 PRINT"
107 PRINT"
108 PRINT"
109 PRINT"
110 PRINT"
111 PRINT"
112 PRINT"
113 PRINT"
114 PRINT"
115 PRINT"
116 PRINT"
120 GOTO120
```

Après chaque **No de ligne,** écris: PRINT" et tape les touches suivantes pour obtenir le **Fleur de lis** Tu peux commencer le dessin à la colonne 20 de l'écran.

101 20 esp., C N, C H
102 19 esp.,RVSon,C K,RVSoff,C K
103 19 esp.,RVSon,1 esp.,RVSoff
104 18 esp.,RVSon,C K,2 esp.,RVSoff,C K
105 18 esp.,RVSon,4 esp.,RVSoff
106 17 esp.,RVSon,C K,4 esp.,RVSoff,C K
107 17 esp.,RVSon,C K,4 esp.,RVSoff,C K
108 18 esp.,RVSon,4 esp.,RVSoff
109 15 esp.,C D,C I,1 esp.,RVSon,C K,2 esp.,RVSoff,C K,1 esp., C I,C F
110 14 esp.,RVSon,C V,3 esp.,RVSoff,1 esp.,RVSon,2 esp.,RVSoff, 1 esp.,RVSon,3 esp.,C C,RVSoff
111 13 esp.,RVSon,C V,4 esp.,RVSoff,C K,RVSon,C K,RVSoff,C K, RVSon,C K,4 esp.,C C,RVSoff
112 13 esp.,RVSon,C F,3 esp.,C I,C N,RVSoff,C L,C J,RVSon,C H, C O,3 esp.,C D,RVSoff

113 14 esp.,RVSon,C F,1 esp.,C D,RVSoff,1 esp.,RVSon,1 esp., RVSoff,C N,C H,RVSon,1 esp.,RVSoff,1 esp.,RVSon,C F,1 esp., C D,RVSoff
114 15 esp.,RVSon,C I,RVSoff,1 esp.,RVSon,C I 2 fois,2 esp., C I 2 fois,RVSoff,1 esp.,RVSon,C I,RVSoff
115 18 esp.,RVSon,C V,2 esp.,C C,RVSoff
116 19 esp.,RVSon,C F,C D,RVSoff

Les caractères graphiques et les caractères de commande sont parfois difficiles à identifier dans un listing fourni par certaines imprimantes, par exemple, le listing du Fleur de Lis de la page précédente...

L'imprimante à aiguilles VIC-1525 marque des points dans une matrice 6 X 7:

Les caractères alphanumériques et quelques autres sont séparés entre eux par une colonne libre Ce qui n'est pas le cas pour les caractères de commande et les caractères graphiques.

Il devient alors difficile de construire un dessin à partir de son listing car des caractères comme C M et C N présentent un aspect identique lorsqu'ils sont imprimés avec la 1525.

Sur l'écran, il n'y a pas de problème car les points sont disposés sur une matrice 8 X 8 dont 2 colonnes et une rangée sont laissées libres pour séparer les caractères.

Dans le cas du listing, on peut faciliter sa lecture en utilisant la fonction CHR$( ). Voici quelques exemples:

DROITE

On retrouve l'image du caractère de droite.

C	M	CHR$(167)
C	N	CHR$(170)
C	L	CHR$(182)
C	K inversé	CHR$(18) CHR$(161)

(CHR$(18) = RVSon)

GAUCHE

C	G	CHR$(165)
C	H	CHR$(180)
C	J	CHR$(181)
C	K	CHR$(161)

EXERCICE 1 Fais l'expérience: Remplace les lignes 101 et 102 par

```
101 PRINT SPC(19)CHR$(170)CHR$(180)
102 PRINT SPC(18)CHR$(18)CHR$(161)CHR$(146)CHR$(161)
```

NOTE a) La fonction SPC(19) déplace le curseur de 19 espaces vers la droite. Donc le caractère suivant s'imprimera à la position 19+1

b) Si tu utilises plusieurs fois la même fonction avec le même argument, il est avantageux (gain de mémoire) de la remplacer par une variable. **Exemple: E$=CHR$(32).**

c) **Depuis la page 124 la liste des programmes provient de** l'imprimante C 4023 (matrice 8X8)

EXERCICE 2 Tu peux sauver plusieurs emplacements mémoires en condensant les lignes 101 à 116 sur 5 ou 6 lignes BASIC. **Fais l'expérience et compare tes résultats** aux lignes 2 à 7 des programmes "GOBE-SOUS" p. 178 et "LA QUOTIDIENNE" page 176.

EXERCICE 3 Dessine un cierge allumé. (**Utilise une feuille** quadrillée comme celle de la page 70).

EXERCICE 4 Complète l'exercice 3 pour obtenir une flamme vacillante...



LOTO : La quotidienne, pages 176 et 177.

TABLEAUX A DEUX DIMENSIONS (en mathématique: matrice)

Un tableau à deux dimensions est un tableau comportant **des rangées et des colonnes.**

Exemple:

	colonne 0	colonne 1	colonne 2
rangée 0			
rangée 1			
rangée 2			
rangée 3			

Rappelle-toi! quand l'ordinateur "fait ses comptes", il commence toujours par <u>zéro</u>.

Notre tableau ci-dessus a 4 rangées (0-3) et 3 colonnes (0-2). Donnons-lui **un nom:** étant donné que ce tableau contiendra tes **notes** scolaires, nous allons utiliser une variable **numérique**, disons qu'il s'appellera C...Mais comme ce tableau possède 12 cases disposées sur **deux dimensions**, il faudra **ajouter deux indices à notre variable C.**

Prenons I pour les rangées et J pour les colonnes et nous aurons C(I,J). Les **indices** (ou index de rangée et index de colonne) permettent de localiser rapidement un des 12 éléments de notre tableau.

Exemple: soit I=3 et J=2, nous aurons C(3,2) c'est-à-dire l'adresse de la case située sur la <u>rangée 3</u>, <u>colonne 2</u> (coin d'en bas, à droite).

Si tu veux mettre la note 86 sur la rangée 1 colonne 2, tu écris: C(1,2)=86. N'oublie pas que le signe = n'a pas le même sens que dans une équation algébrique; dans un programme BASIC il faut lire cette égalité comme suit: **mettre 86 dans la variable C(1,2).**

<u>**EXERCICE 5**</u> Construis un petit programme qui affichera un tableau des notes scolaires suivantes:

	MATH.	FRANC.	HIST.
Première semaine	82	75	92
Deuxième semaine	70	65	86
Troisième semaine	78	68	75
Quatrième semaine	90	80	85

<u>**PROGRAMME** (première version):</u>

```
5 DIM C(3,2)
10 FOR I=0 TO 3: FOR J=0 TO 2
15 INPUT"NOTES";N
20 C(I,J)=N
25 NEXT J: NEXT I
```

Tu écris RUN et tu entres (INPUT) les notes. Lorsque tu inscris la 12e note et que tu appuies RETURN, l'écran affiche READY. **N'écris pas RUN** car la commande RUN mettra à zéro tous les éléments de ton tableau (n'oublie pas que ton programme ne contient pas encore d'instruction PRINT!).

Ecris en mode direct: PRINT C(0,0) RETURN. L'écran affiche 82 c'est-à-dire l'élément de la rangée 0 colonne 0. Fais d'autres essais... PRINT C(1,2)... etc.

EXERCICE 6 Ajoute les instructions suivantes pour afficher automatiquement tous les éléments de ton tableau:

```
40 FOR I=0 TO 3: FOR J=0 TO 2
50 PRINT C(I,J),
60 IF J=2 THEN PRINT"███"
70 NEXT J: NEXT I
```

La [,] à la fin de la ligne BASIC No 50 fait imprimer les résultats sur 4 sections de l'écran. La ligne BASIC No 60 fait descendre le curseur de trois rangées dès que le C64 a imprimé les 3 éléments d'une rangée (colonnes 1, 11, 21).

EXERCICE 7 Dans le programme **ci-dessus**, tu devais entrer les notes(INPUT) chaque fois que tu écrivais RUN. Pour conserver tes données, inscris les notes(constantes) dans une instruction DATA et mets READ N dans une ligne No15. Tu auras cette version:

PROGRAMME (deuxième version)

```
5 DIM C(3,2)
10 FOR I=0 TO 3: FOR J=0 TO 2
15 READ N
20 C(I,J)=N
25 NEXT J: NEXT I
30 PRINT"█"
40 FOR I=0 TO 3: FOR J=0 TO 2
50 PRINT C(I,J),
60 IF J=2 THEN PRINT"███"
70 NEXT J: NEXT I
80 DATA 82,75,92,70,86,65,78,68,75,90,80,85
```

Fais RUN et RETURN. Avant de passer à l'exercice 10, écris en mode direct: **PRINT FRE(0)+65536.** L'écran affiche le **nombre d'octets disponibles:** 38625. Ecris cette valeur sur un bout de papier, tu en auras besoin à l'exercice 10.

VARIABLES ENTIERES : (symbole: %)

EXERCICE 8 Insère (INST) **le symbole %** entre la variable C et la parenthèse aux lignes 5, 20 et 50 de l'exercice précédent.

```
5 DIM C%(3,2)
20 C%(I,J)=N
50 PRINT C%(I,J);
```

Fais RUN et RETURN et écris en mode direct: **PRINT FRE(0)+65536. L'écran affiche** 38658 (38625 +33). Donc en utilisant une **variable entière,** tu fais un gain de 3 octets (bytes) pour chaque élément du tableau. Comme il y a 12 éléments dans ton tableau, **tu économises 33 octets,** c'est-à-dire 36 moins les 3 caractères % que tu as insérés dans ton programme.

Les variables entières (avec %) offrent surtout un intérêt "économique"... (un gain de place en mémoire). On les utilise surtout pour **définir des indices de listes ou de tableaux** car ces indices sont obligatoirement des valeurs entières. **Application pratique:** Dans le programme de LOTO "La Quotidienne" page 176 tu remarqueras l'emploi d'une variable entière pour définir les indices du tableau LT%(I,J). Un gain de 120 octets (moins 3)!...

QUESTION D'où proviennent les 3 octets économisés sur chaque élément du tableau?

Réponse: Le C64 accepte 3 types de variables: 1- type entier (avec %), 2- type réel, 3- type chaîne (avec $). Le C64 considère comme réelle une variable qui n'est pas accompagnée de % où de $. Une **variable réelle demande 5 octets** (format virgule flottante)... et une variable entière en demande 2... d'où cette économie de 3 octets (explications supplémentaires: vol.2).

NOTE La présence d'une liste ou d'un tableau dans un programme **ralentit son exécution** parce que les tableaux occupent la zone mémoire située au-dessus des variables et **chaque fois** que le déroulement du programme introduit une nouvelle variable, **toute la zone réservée à ton tableau doit être déplacée**!

CONCLUSION: Lorsqu'un programme contient un tableau, l'ordinateur lui réserve une "zone" spéciale dans ses emplacements mémoires; c'est pour cette raison qu'on appelle parfois **un tableau: une zone.** Les éléments d'un tableau (zone) se trouvent ainsi juxtaposés et les adresses des éléments du tableau forment une succession de variables équivalentes.

Quand tu auras appris à manipuler les fichiers (volume II), tu verras que les entrées (INPUT) des notes qui forment ton tableau pourront être **conservées dans un fichier.** Exemple: **Fichier NOTES.**

LES ENTREES CLAVIER

De quelles façons l'ordinateur peut-il **lire les** <u>données venant du clavier pendant l'exécution</u> d'un programme?

<u>Les trois principaux moyens</u> qui permettent une "entrée clavier" **pendant l'exécution d'un programme** sont les deux instructions INPUT et GET et la fonction PEEK.

1- INPUT (voir pages 125 à 128)
1- GET (voir pages 45 et 46)
3- PEEK(197) (ou peek (203))

Etudions ce troisième cas. Rappelle-toi d'abord que PEEK est une fonction et que les fonctions sont toujours suivies de parenthèses entre lesquelles tu mets l'argument.

Avec la fonction PEEK( ) le **<u>nombre entre les parenthèses</u>** <u>est toujours **une des 65535 mémoires** du C64.</u>

Chaque fois que tu appuies une des touches du clavier, la mémoire 197 (ou 203) **enregistre le numéro de cette touche**

<u>EXERCICE 9</u> Ecris en mode programmé:

```
10 PRINT PEEK(197)              RUN et RETURN
```

L'écran affiche 1. **C'est le numéro de la touche RETURN** (la dernière touche que tu as tapée).

<u>EXERCICE 10</u> Ajoute cette ligne 5 à l'exercice précédent:

```
5 IF PEEK(197) <> 39 THEN 5
10 PRINT PEEK(197)              RUN et RETURN
```

Le C64 **attend** que tu tapes la **touche numéro 39 (la lettre N).** Les autres touches sont inopérantes (sauf "stop" qui arrête le programme).

Si tu appuies N, le C64 sort de la boucle et imprime le numéro de la touche N: 39. Le caractère N s'imprimera aussi mais après READY... et si tu as tapé d'autres caractères avant N, ils seront aussi affichés après READY...mais il y aura un maximum de 10 caractères affichés car le tampon clavier possède 10 mémoires (631 à 640).

CONCLUSION: L'instruction de la ligne 5 ci-dessus peut donc jouer <u>le même rôle que GET accompagné de IF...THEN</u>.

<u>Expérience:</u> Introduis ton programme A-2 de la page 58 dans ton ordinateur. Remplace l'instruction GET des lignes 1225, 1235, 1245 et 1345 par la fonction PEEK(197). Exemple:

```
1225 GET N$: IF N$ <> "N" THEN 1225
```

Remplace cette ligne 1225 par:
1225 IF PEEK(197) <> 39 THEN 1225

Cette instruction produira le même effet que la précédente. Tu peux faire les mêmes transformations pour:
—allumer la lampe L (à 1235); —produire l'éclair E (à 1245)
—revenir au départ D (à 1345)

Examine le tableau ci-dessous et tu connaîtras les numéros des lettres L, E, D.

<u>TABLEAU</u>(ou matrice) **<u>des NOS des touches du clavier:</u>**

	127	191	223	239	247	251	253	254
127	STOP 63	/ 55	, 47	N 39	V 31	X 23	SHIFT gauche 15	CRSR 7
191	Q 62	↑ 54	@ 46	O 38	U 30	T 22	E 14	f5 6
223	C= 61	= 53	: 45	K 37	H 29	F 21	S 13	f3 5
230	ESPACE 60	SHIFT droite 52	. 44	M 36	B 28	C 20	Z 12	f1 4
247	2 59	HOME 51	- 43	ZERO 35	8 27	6 19	4 11	f7 3
251	CTRL 58	; 50	L 42	J 34	G 26	D 18	A 10	CRSR 2
253	← 57	* 49	P 41	I 33	Y 25	R 17	W 9	RETURN 1
254	1 56	£ 48	+ 40	9 32	7 24	5 16	3 8	DEL Ø

Le tableau à 2 dimensions(**ou matrice**) ci-dessus a été obtenu au moyen d'un petit programme en langage machine (vol. 2)

Cependant tu peux obtenir <u>le No</u> de la plupart <u>des touches</u> si tu exécutes ces trois lignes BASIC:

```
10 FOR T=1 TO 100: NEXT: A=PEEK(197): IF A=64 THEN 10
20 PRINT PEEK(197),
30 GOTO 10
```

APPLICATION: ligne 220 page 176

LES FONCTIONS TRIGONOMETRIQUES : SIN, COS, TAN, ATN:

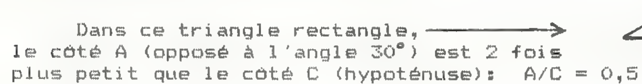
Même si cette page et la suivante te paraissent un peu austères, ne "lache pas!" car ces notions te seront très utiles pour **tracer les courbes représentatives d'une fonction.**

Comme toutes les autres fonctions, les fonctions trigonométriques doivent être suivies de parenthèses entre lesquelles tu mets l'argument.

L'argument de SIN, COS, TAN (SINus, COSinus et TANgente) est la valeur de **l'angle en radians**. L'argument de ATN (Arc TangeNte) est la VALEUR **de la tangente** en radians.

Tu es probablement plus familier avec les degrés. Quand le rayon r d'un cercle fait un tour complet on dit qu'il a tourné
d'un angle de 360 degrés ou
d'un angle 2 π radians (6,28 radians).

360 degrés = 2 π radians
Donc 1 degré = π/180 radians
et 30 degrés = 30* π/180 radians



Dans ce triangle rectangle, →
le côté A (opposé à l'angle 30°) est 2 fois plus petit que le côté C (hypoténuse): A/C = 0,5

On dit: SINUS 30 degrés = 0,5

EXERCICE 11 Ecris en mode direct:

```
PRINT SIN(30* π/180)              RETURN.  L'écran affiche .5
```

Tu as multiplié 30 degrés par π/180 pour transformer les degrés en radians car BASIC travaille avec des radians!

Dans ce triangle rectangle →
le côté B (adjacent à l'angle 60°) est 2 fois plus petit que le côté C: B/C = 0,5

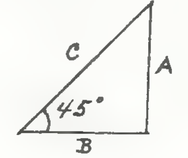
On dit: COSINUS 60 degrés = 0,5

EXERCICE 12 Ecris en mode direct:

```
PRINT COS(60* π/180)              RETURN.  L'écran affiche .5
```

Donc sinus 30° = cosinus 60°

Dans ce triangle rectangle, →
les côtés A et B sont égaux: A/B = 1



On dit: TANGENTE 45 degrés = 1

EXERCICE 13 Ecris en mode direct:

PRINT TAN(45* π /180) RETURN. L'écran affiche .999999999

On devrait obtenir 1. C'est un cas "d'arrondis" dû à la valeur approchée de π

Avec un angle de 30°, tangente 30° = A/B = 0,577350269 soit environ 0,58.

EXERCICE 14 Ecris en mode direct:

PRINT TAN(30* π /180) RETURN. L'écran affiche .577350269

La fonction ATN (arc tangente) est l'**inverse** de la fonction TAN

EXERCICE 15:

PRINT ATN (.577350269)*180/π RETURN. L'écran affiche 30 (degrés)

L'argument est la VALEUR de la tangente en radians

180/π degrés/radian pour obtenir la réponse en degrés.

BASIC dispose d'une seule fonction trigonométrique inverse: ATN. Quoi faire alors si on a besoin de l'arc cosinus ou de l'arc sinus? Nous allons voir que BASIC dispose d'une instruction qui permet aux programmeurs de définir ses propres fonctions. Tu pourras alors mieux comprendre le programme de la page 168 **qui calcule les angles d'un triangle quelconque.**

DEF FN:

L'instruction DEF FN permet aux programmeurs de DEFinir ses propres FoNctions.

Le nom de la fonction sera formé par FN suivi de la variable numérique que tu choisiras et des parenthèses contenant l'argument muet de la fonction; tu mets ensuite le signe = et la formule appropriée.

1- Définition d'une fonction "arrondi": AR(X)
DEF FNAR(X) = INT(X*10+.5)/10 (arrondi à 1 décimale)

2- Définition d'une fonction sinus qui transforme les degrés en radians:
SI(X)
DEF FNSI(X) = SIN(X* π /180)

3- Définition d'une fonction <u>arc cosinus:</u> AC(X)
DEF FNAC(X) = -ATN(X/SQR(-X*X+1)+ π/2

Il ne faut pas oublier que ces instructions DEF servent uniquement à **définir** une fonction. L'argument X est un argument formel (muet) qui prendra une valeur **seulement si** la fonction **est appelée** ailleurs dans le programme.

Tu peux appeler (évaluer) une de ces fonctions en utilisant son nom. L'argument prendra la valeur **spécifiée dans l'appel** et non dans la DEFinition.

NOTE Les sous-programmes **ressemblent** aux fonctions définies par le programmeur en ce sens qu'ils doivent **aussi être appelés** à partir d'une ligne quelconque du programme.

<u>EXERCICE 16</u> Voici un programme qui contient des instructions DEF FN pour <u>calculer les angles</u> d'un **triangle quelconque:**

```
1 REM CALCUL DES ANGLES D'UN TRIANGLE QUELCONQUE
3 REM DEFINITION D'UNE FONCTION 'ARC COS.' EN DEGRES
5 DEF FNAC(X)=(-ATN(X/SQR(-X*X+1))+π/2)*180/π
7 DEF FNAR(X)=INT(X*10+.5)/10: REM 'ARRONDI A 1 DECIMAL'
10 PRINT "VALEURS A1,B1,C1  DES COTES DU TRIANGLE?"
15 PRINT"(C1 EST LE PLUS GRAND)"
20 INPUT A1,B1,C1
25 IF C1<A1 OR C1<B1 THEN 10
30 CA=(B1↑2+C1↑2-A1↑2)/(2*B1*C1): REM CA←COSINUS A
40 CB=(A1↑2+C1↑2-B1↑2)/(2*A1*C1): REM CB←COSINUS B
45 IF CA=>1 OR CB=>1 THEN PRINT"PAS DE TRIANGLE": END
50 A=FNAC(CA): REM L'ARC COS. DE LA'VALEUR'DU COS A DONNE L'ANGLE A
60 B=FNAC(CB): REM L'ARC COS. DE LA'VALEUR'DU COS B DONNE L'ANGLE B
70 C=180-A-B
80 PRINT "A1=";A1;"B1=";B1;"C1=";C1
85 PRINT"ANGLE OPPOSE AU COTE A1: A=";FNAR(A);"DEGRES
90 PRINT"ANGLE OPPOSE AU COTE B1: B=";FNAR(B);"DEGRES
95 PRINT"ANGLE OPPOSE AU COTE C1: C=";FNAR(C);"DEGRES
```

Ecris RUN et tape RETURN. Le C64 attend l'entrée (INPUT) des valeurs de chacun des 3 côtés du triangle.

Exemple 1: Les côtés: A1=3 B1=4 C1=5.
L'écran affiche les angles en degrés:
Les angles: A=36.9 degrés B=53.1 degrés C=90 degrés
Les résultats sont arrondis à <u>une décimale</u> et <u>l'ordinateur</u> sépare les entiers des décimales <u>par un point</u>
Exemple 2: Les côtés: A1=25.2 B1=37.8 C1=43.4.
Les angles: A=35.3 degrés B=60.1 degrés C=84.6 degrés
Exemple 3: Les côtés: A1=2 B1=2 C1=4
L'écran affiche: Pas de triangle.

La formule de base utilisée pour l'exercice 16 est celle de la loi des cosinus: $a^2 = b^2 + c^2 - 2bc\cos A$.

Donc $\cos A = \dfrac{b^2 + c^2 - a^2}{2bc}$ et $A = \text{arc cosinus } \dfrac{(b^2 + c^2 - a^2)}{2bc}$

Etant donné que l'arc cosinus n'existe pas en BASIC, nous avons **DEF**ini cette **FoN**ction et mis les degrés en radians à la ligne 5. Une fonction "arrondi" a été définie à la ligne 7. NOTE: Il y a d'autres algorithmes qui permettent de résoudre le problème en utilisant des **relations trigonométriques**.

Le tracé de la courbe représentative d'une fonction :

La fonction sinus nous permet de tracer une sinusoïde ou un mouvement sinusoïdal sur l'écran.

EXERCICE 17 Utilise les fonctions SIN et TAB pour générer un mouvement sinusoïdal avec le caractère étoile de couleur pourpre. Départ à la 11ème colonne de l'écran

```
5 REM 9 EST LE FACTEUR D'ECHELLE DE LA FONCTION SINUS
10 Y=9*SIN(A*π/180)+11: REM +11 POUR DEPART COLONNE 11 (SINUS A=0)
20 A=A+20: REM L'ANGLE A AUGMENTE DE 20 DEGRES A CHAQUE ITERATION
30 PRINTTAB(Y)"█*"
40 FORT=1TO150:NEXT
50 GOTO10
```

EXERCICE 18 Ajoute une fonction SIN négative pour former une deuxième sinusoïde opposée à la première avec départ sur la colonne 29. Couleur jaune pour la première sinusoïde et blanche pour la deuxième.

```
5 REM 9 EST LE FACTEUR D'ECHELLE (VOIR P.170,AU MILIEU)
10 Y=9*SIN(A*π/180)+11: REM DEPART SUR COLONNE 11 (ANGLE A=0)
15 Y1=-9*SIN(A*π/180)+29: REM DEPART COLONNE 29 (VERS LA GAUCHE)
20 A=A+20: REM ANGLE A AUGMENTE DE 20 DEGRES (SCHEMA P.170)
30 PRINTTAB(Y)"█*█"
35 PRINT TAB(Y1)"□*"
40 FORT=1TO150:NEXT
50 GOTO10
```

EXERCICE 19 Remplace l'étoile de l'exercice 17 par un oiseau ayant les ailes blanches et le corps jaune. La sinusoïde devra occuper 38 colonnes d'écran.

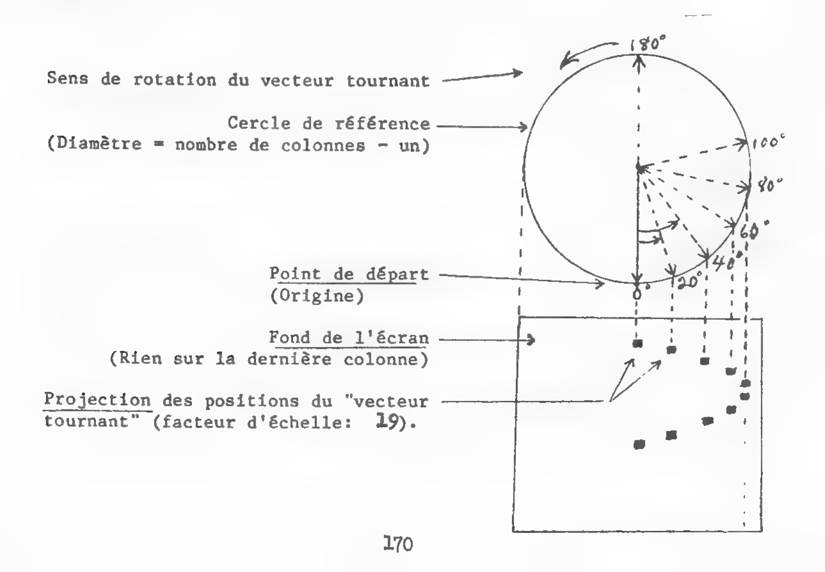
```
5 REM 18 EST LE FACTEUR D'ECHELLE DE LA FONCTION SINUS
10 Y=18*SIN(A*π/180)+19: REM +19 POUR DEPART COLONNE 19 (SINUS A=0)
20 A=A+20: REM L'ANGLE A AUGMENTE DE 20 DEGRES
30 PRINTTAB(Y)"█╱█●█╲████";: FOR T=1 TO 75: NEXT T
40 PRINTTAB(Y)"╲█●█╱": FOR T=1 TO 75: NEXT
50 GOTO10
```

<u>EXERCICE 20</u> Remplace l'oiseau par un skieur qui fait du slalom. Fais tourner les skis vers la droite ou vers la gauche selon que le skieur se dirige vers la droite ou la gauche de l'écran. Ajoute des petits drapeaux là où la sinusoide atteint son amplitude maximale. <u>Note</u>: Un dessin symbolique du skieur: \o\ et /o/ Tu feras mieux (volume 2) avec la programmation des caractères

```
5 PRINT"♥
10 FOR I=1 TO 18: REM 360 DEGRES/20 = 18 (18 DESSINS PAR CYCLE)
20 A=18*SIN(B*π/180)+19: REM *π/180 MET LES DEGRES EN RADIANS(P.166)
30 B=B+20
40 IF I=15 THEN PRINT"▯▮▮▮▙"SPC(39)"| "
45 IF I=6 THEN PRINT TAB(35)"▯▖"SPC(39)" |"
50 IF I>5 AND I<15 THEN PRINT TAB(A)"/O/" SPC(38)"    "
55 IF I<6 OR I>14 THEN PRINT TAB(A)"\O\" SPC(37)"    "
60 FOR T=1 TO 150: NEXT T
65 NEXT I
70 GOTO 10: REM RECOMMENCER LA BOUCLE
```

Les 4 exercices précédents illustrent un mouvement sinusoidal avec un axe <u>oy horizontal</u> (orienté vers la droite de l'écran) et un axe <u>ox vertical</u> **(abscisse orientée vers le bas)**, les coordonnées (0,0) de l'origine se trouvant sur la onzième colonne de l'écran.

Etant donné que la valeur du sinus varie entre **moins un et plus un**, il faut multiplier sa valeur par un **facteur d'échelle** (ou coefficient d'affinité) qui donnera à la sinusoide une amplitude maximale de + ou − 19. (Moins 20 et plus 20 provoquerait un retour chariot à l'écran du C64).



WAIT

En plus des instructions INPUT et GET et de la fonction PEEK(197), **il y a aussi l'instruction WAIT qui permet de suspendre l'exécution d'un programme** en attendant qu'une touche soit appuyée.

Exemple: L'adresse 653 contient normalement la valeur 0. Si tu presses la touche SHIFT, la valeur un s'inscrit dans la mémoire 653. Si tu relaches SHIFT, la mémoire 653 reprend la valeur 0.

Mémoire 653 (état normal)

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Tous les bits sont à zéro |

EXERCICE 21 Ecris une instruction qui arrête l'exécution du programme jusqu'à ce que la touche SHIFT soit tapée

WAIT 653,1,0 ou mieux WAIT 653,1 (car le 2e paramètre **n'est pas obligatoire quand il vaut zéro**)

Application: Voir ligne 58 du programme: "GOBE-SOUS" page **178**

Mémoire 653 quand SHIFT est appuyée:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | WAIT 653,1 |

1 AND 1

Le C64 fait le ET (AND) logique entre le profil binaire de la mémoire 653 et le paramètre (ici, 1). 1 AND 1 = 1 donc le programme continue et exécute l'instruction qui suit WAIT puisque le résultat est 1.

Le rôle de WAIT est de tester l'état d'un bit (0 ou 1). Un bit est une unité d'information (anglais: BInary digiT). Un octet (anglais: byte) contient 8 bits. Chaque bit peut prendre la valeur 1 ou 0.

QUESTION Quelles touches attendra l'ordinateur si tu écris:
WAIT 653,7

Réponse: Le C64 fera **le ET logique (AND)** entre le profil binaire du contenu de la mémoire 653 et la valeur binaire de 7.

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

ATTENTE

Le C64 attendra l'une ou l'autre des 3 touches: SHIFT ou C= ou CRTL.

Si tu appuies sur SHIFT, le premier bit (dans 653) prend **l'état 1: 00000001**

Si tu appuies sur C=, le deuxième bit (dans 653) prend **l'état 1:** 00000010

Si tu appuies sur CTRL, le troisième bit (dans 653) prend **l'état 1: 00000100**

Puisque 1 AND 1 = 1, l'ATTENTE SERA TERMINEE si tu appuies **l'une des** 3 touches signalée ci-dessus.

TABLE de VERITE (Boole)

1 AND 1 = 1	1 OR 1 = 1	NOT 1 = 0
0 AND 1 = 0	0 OR 1 = 1	NOT 0 = 1
1 AND 0 = 0	1 OR 0 = 1	
0 AND 0 = 0	0 OR 0 = 0	

Si tu mets un 2e paramètre, = < 7 alors l'instruction WAIT perdra son "pouvoir" d'attente parce que, **pour ce 2e paramètre, le C64 fait le OU logique (OR)** exclusif.

NOTES:
1- La touche STOP ne peut interrompre l'attente de l'instruction WAIT
2- L'attente de la touche SHIFT est intéressante parce qu'on n'a pas besoin de vider le tampon clavier (les caractères appuyés ne s'impriment pas après READY).
3- Ne pas confondre le **numéro** d'une touche et le **code** de son caractère. Ex: le caractère N: **touche no 39** (Tab. p. 165) et **code 14 avec POKE** (Tab. p. 75) ou **code 78 avec PRINT...** (Tab. des carac. ASCII p. 76). Ex. PRINT ASC("N") RETURN. L'écran affiche 78.

Les fonctions RIGHT$, LEFT$, MID$

Les fonctions RIGHT$, LEFT$ et MID$ sont des fonctions d'extraction de caractères. Le traitement de texte ou traitement de chaînes de caractères est rendu possible grâce à ces 3 fonctions associées à la fonction LEN étudiée au chapitre IV. Rappelle-toi qu'une fonction est toujours suivie de parenthèses entre lesquelles tu mets l'argument (ou paramètre).

1- La fonction RIGHT$ (droite) exige 2 paramètres: le premier est une chaîne de caractères (ou une variable chaîne), le deuxième un nombre entier qui peut être une variable ou une constante.

EXERCICE 22 Ecris en mode direct:

```
PRINT RIGHT$ ("JEANNE DUPONT",6)
```

L'écran affiche DUPONT. Cette instruction PRINT doit se lire ainsi: **A partir de la droite** de la **chaîne "JEANNE DUPONT", extraire 6** caractères.

A l'exercice 22, mets 7 au lieu de 6 et l'écran affichera un espace devant DUPONT. **Fais d'autres expériences** et redis chaque fois la phrase: **A partir de....de la chaîne....extraire...**

2- **La fonction LEFT$** (gauche) exige aussi 2 paramètres.

EXERCICE 23 Ecris en mode direct:

```
PRINT LEFT$ ("JEANNE DUPONT",6)
```

L'écran affiche JEANNE. Répète cette phrase: A partir de la gauche de la chaîne "JEANNE DUPONT", extraire 6 caractères. Remplace 6 par 8 et l'écran affichera un espace et la lettre D après JEANNE. **Fais d'autres expériences.**

3- **La fonction MID$** (anglais: MIDle=milieu) a trois paramètres: le premier est une chaîne de caractères (ou variable chaîne), le deuxième est un nombre entier qui indique **à partir de quel** caractère l'extraction **doit commencer**, le troisième est **le nombre de caractères à extraire.**

EXERCICE 24 Ecris en mode direct:

```
PRINT MID$ ("LE MONDE MERVEILLEUX DE W.D.",13,8)
```

L'écran affiche VEILLEUX. Tu vois que cette fonction MID$ est générale et peut être utilisée à la place des deux fonctions DROITE et GAUCHE. Redis la phrase: **A partir du 13e** caractère de la **chaîne** "LE MONDE MERVEILLEUX DE W.D." **extraire** 8...

NOTE Le 3e paramètre n'est **pas obligatoire** si tu veux extraire tous les caractères à partir du caractère indiqué dans le 2e paramètre. Fais l'expérience, enlève 8 et l'écran affichera VEILLEUX DE W.D.

CONCATENATION: Tu peux faire la concaténation (addition) de plusieurs chaînes de caractères ou extraire certains caractères d'une chaîne ou de plusieurs chaînes **pour former une nouvelle chaîne.** Utilise les fonctions d'extraction pour résoudre le prochain exercice.

EXERCICE 25 De la chaîne: **"Le no de Tél. du LABO est 357-8215 et le code rég. 819"** extrais le code rég. et le No de tél. Mets-les ensuite dans une nouvelle chaîne.

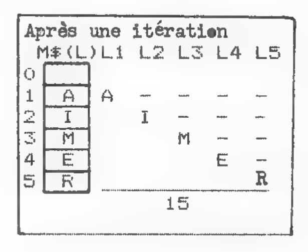
```
10 T$="LE NO DE TEL. DU LABO EST 357-8215 ET LE CODE REG. 819"
20 T1$=MID$(T$,10,3)
30 T2$=RIGHT$(T$,4)
40 T3$=MID$(T$,26,9)
50 PRINT+T1$+T2$;T3$
```

L'écran affiche: TEL 819 357-8215. A la ligne 40, la fonction MID$ extrait 9 caractères à partir du caractère espace (26e) afin de séparer le code du no de tél. dans la nouvelle chaîne. La ligne 50 fait la concaténation (addition) des 3 chaînes extraites aux lignes 20 à 40.

EXERCICE 26 Un anagramme est un mot formé par la transposition des lettres d'un autre mot. Ainsi l'anagramme de AIMER est MARIE et celui de RAGE est GARE. Construis un programme qui formera le mot MARIE en transposant les lettres du mot AIMER. Tu fais arrêter l'exécution du programme aussitôt que le mot MARIE est formé.

```
10 MA$="MARIE"
20 FOR L=1 TO 5:M$(L)=MID$(MA$,L,1):NEXT
30 FOR L1=1 TO 5
40 FOR L2=1 TO 5: IF L2=L1 THEN 122
50 FOR L3=1 TO 5: IF L3=L1 THEN 121
60 IF L3=L2 THEN 121
70 FOR L4=1 TO 5: IF L4=L1 THEN 120
80 IF L4=L2 THEN 120
90 IF L4=L3 THEN 120
100 L5=15-(L1+L2+L3+L4)
110 PRINT M$(L1);M$(L2);M$(L3);M$(L4);M$(L5)
115 PRINT: IF L1=2 AND L2=4 AND L3=1 AND L4=5 AND L5=3 THEN END
120 NEXT L4
121 NEXT L3
122 NEXT L2,L1
```

Le programme contient 4 boucles FOR...NEXT imbriquées. La boucle extérieure donne la valeur L1, la boucle suivante donne L2, etc. Ces variables L1,L2,...prendront toutes les valeurs possibles à la condition qu'elles soient différentes. Quand tu connais L1, L2, L3, L4, tu connais aussi L5: L5 = 15 - (L1 + L2 + L3 + L4)



EXERCICE 27 Un palindrome est un mot ou une phrase qu'on peut lire dans les deux sens. La chaîne: **"Léon a trop par rapport à Noel"** est un palindrome. Fais un programme qui écrira la chaîne (inversé)

```
10 PA$="LEON A TROP PAR RAPPORT A NOEL"
20 L=LEN(PA$)
30 P$=""
40 FOR I=1 TO L
50 PA$=LEFT$(PA$,L-I+1)
60 A$=RIGHT$(PA$,1)
70 P$=P$+A$
80 NEXT
90 PRINT P$
95 REM AJOUTER 65 IF I =19 THEN 80
96 REM AJOUTER 75 IF I=11 THEN P$=P$+" "
```

Pour que les caractères espaces coincident il faut ajouter les 2 lignes suivantes:
65 IF I=19 THEN 80
75 IF I=11 THEN P$=P$+" "

<u>EXERCICE 28</u> **COMPARAISON.** Ecris un programme de <u>traduction</u> francais-anglais. Mets les mots dans une instruction DATA (expérience avec 5 mots français et 5 mots anglais).

```
10 FOR I=1 TO 5: READ MF$(I),MA$(I): NEXT
20 PRINT"[illegible]MOT FR. A TRADUIRE:":INPUT F1$
30 L1=LEN(F1$)
40 FOR I=1 TO 5: F$="":A$=""
50 FF$=MF$(I)
60 L2=LEN(FF$): EG$=LEFT$(FF$,1)
70 FOR J=1 TO L1
80 EM$=MID$(F1$,J,1): IF EM$<>EG$ THEN 170
90 MM$=MID$(F1$,J,L2): IF MM$<>FF$ THEN 170
100 IF J-1=0 THEN 120
110 F$=LEFT$(F1$,J-1)
120 IF L1-J-L2+1=0 THEN 140
130 ED$=RIGHT$(F1$,L1-J-L2+1)
140 F1$=F$+MA$(I)+ED$
150 L3=LEN(MA$(I)): L1=L1-L2+L3
160 F$="":A$=""
170 NEXT J,I
180 PRINT:PRINT F1$: GOTO 20
200 DATA TETE,HEAD,CORPS,BODY,BRAS,ARM,JAMBE,LEG,PIED,FOOT
```

> Ce petit progr. n'accepte pas les caractères [,] et [:] Tu peux le compléter avec des instruct. incluant GET.

> **On dit que la chaîne "CORPS" est plus petite que** la chaîne "TETE" parce que le code **C** précède le code **T...Voir tabl. page 76.**

<u>EXERCICE 29</u> Reprends l'exercice 26-2 p.141 et organise le programmme pour que le bulletin de nouvelles se déplace **de la droite vers la gauche à l'écran.** Tu peux ajouter le bruit des frappes d'une machine à écrire:

```
10 PRINT"[illegible]NOUVELLES EXPRESS[illegible]"
20 A$="***APPARITION D'UNE SOUCOUPE VOLANTE   "
25 B$="LES EXTRATERRESTRES SONT LA!    "
30 C$="RESTEZ SUR TERRE***                    "
40 D$="PROCHAIN BULLETIN A 9H"
50 G$=A$:GOSUB 100
60 G$=B$: GOSUB 100
65 G$=C$: GOSUB 100
70 PRINT"[illegible]"D$: END
100 FOR L=1 TO LEN(G$)
110 PRINT"[illegible]" MID$(G$,L,1);: GOSUB 200: PRINT"[illegible]";
120 NEXT L: RETURN
200 FOR T=1 TO 100: NEXT: RETURN
```

↑ 40 espaces

> Le caractère [T] inversé (DEL) fait reculer le curseur d'une case et supprime le caractère qui s'y trouvait <u>(No 9 p35)</u>

(Comment obtenir le [T]? Explications page 180

Dès que le programme de l'exercice 29 a fonctionné 1 fois (RUN RETURN), le **T inversé et le caractère qui le précède disparaissent du listing de l'écran:**

Ils continueront quand même à jouer leur rôle (ex.: le E inversé pour la couleur blanche). **Si tu fais le moindre changement dans la ligne 110, il faudra réécrire les 2 chaînes.** Pour mieux comprendre cette ligne 110, fais en mode direct ce que les 2 chaînes font en mode programmé.

<u>NOTE</u> A la page suivante tu as la **1ère partie** de **La QUOTIDIENNE**...La 2ème partie ainsi que l'adaptation à **La quotidienne 4** paraîtront dans le volume 2

Programme d'une LOTO: "LA QUOTIDIENNE"

```
1 DIM LT%(9,3),L(3),C(3): POKE 53281,1
2 PRINT"■■" TAB(19)"▌":PRINT TAB(19)"■■":PRINT TAB(19)"■ ■"
3 PRINT TAB(18)"■  ■":PRINT TAB(18)"■   ■":PRINT TAB(17)"■    ■"
4 PRINT TAB(17)"■    ■":PRINT TAB(18)"■   ■":PRINT TAB(15)"▬ ■  ■ ▬"
5 PRINT TAB(14)"■  ■ ■ ■ ■  ■":PRINT TAB(13)"■   ■■■■   ■"
6 PRINT TAB(13)"■  ▬■▌■▬  ■":PRINT TAB(14)"■ ■ ■ ■▌■■■ ■"
7 PRINT TAB(15)"■■ ■▬ ▬■ ■■":PRINT TAB(18)"■  ■"
8 PRINT TAB(19)"■ ■":PRINT TAB(14)"■LOTO  QUEBEC■"
9 PRINT TAB(13)"■LA QUOTIDIENNE" SPC(27)"PAR L.PICARD■"
10 PRINT TAB(11)"■■APPUIE  UNE TOUCHE■"
11 GET A$: IF A$="" THEN 11
12 PRINT"■■■■TU SELECTIONNES 4 CHIFFRES QUI FORMERONT"
13 PRINT"UN NOMBRE COMPRIS ENTRE 0000 ET 9999"
14 PRINT"■■■VOIR TABLEAU DES GAINS ■O■ OU ■N■"
15 GET B$: IF B$="O" THEN GOSUB 700:GOTO 18
16 IF B$="N" THEN 18
17 GOTO 15
18 PRINT"■■TAPE■ LE CHIFFRE SELECTIONNE ET": PRINT"■■APPUIE 'RETURN'■"
20 FOR I=0 TO 9
25 FOR J=0 TO 3
27 READ LT%(I,J): PRINT"■■"TAB(14)LT%(I,J);
28 NEXT J: PRINT: IF I=9 THEN 32
30 PRINT: NEXT I
32 RESTORE
33 PRINT SPC(6)"SELECTION";
35 OPEN2,0
36 FOR I=0 TO 3
37 INPUT#2,A: IF A>9 THEN 37
40 L(I)=A
50 PRINT" ";: NEXT I
60 N=144: V=30: R=28: J=158: W=0: X=18: Y=0
70 PRINT"■" SPC(8)"■LOTO■ 'LA QUOTIDIENNE'"
75 REM F1 EST LA PREMIERE DES 4 TOUCHES A DROITE
80 PRINT:PRINT SPC(7)"■TA SELECTION: ";
85 FORI=0TO3:PRINT"■"L(I)"■";:NEXT: PRINT SPC(37)"■■■TAPE F1"
90 PRINT: CLOSE2
100 PRINT"■■■■■■"CHR$(N)"                    ●"
110 PRINTCHR$(V)"              ●"CHR$(N)"   |"CHR$(J)"   ●
120 PRINTCHR$(V)"               \ "CHR$(N)"|"CHR$(J)" /
130 PRINTCHR$(V)"                \"CHR$(N)"|"CHR$(J)"/
140 PRINTCHR$(R)"          ●─────■O"CHR$(R)"────●
160 PRINTCHR$(J)"                /"CHR$(N)"|"CHR$(V)"\"
170 PRINTCHR$(J)"               /"CHR$(N)"  |"CHR$(V)" \
180 PRINTCHR$(J)"              ●"CHR$(N)"   |  "CHR$(V)" ●
190 PRINTCHR$(N)"                   ●■
192 PRINT
195 PRINT SPC(12)CHR$(109)CHR$(32)CHR$(110)
196 FOR I=1 TO 3: PRINT TAB(12)CHR$(167)CHR$(32)CHR$(165): NEXT
197 PRINT TAB(12)CHR$(167): PRINTTAB(13)CHR$(163)
200 FOR T=1 TO 100: NEXT: REM RALENTIR LA ROUE
209 REM K=VARIABLE SERVICE POUR ROTATION COULEURS
210 K=J: J=N: L=R: R=K: M=V: V=L: N=M
215 IF Y=4 THEN GOSUB 600: GOTO250: REM Y=BOULES CHOISIES HASARD
217 IF W>0 THEN GOSUB 500
220 IF PEEK(197)=4  THEN GOSUB 500: REM TAPE F1 POUR AVOIR 4 BOULES
230 PRINT"■■■■■■■■■■■■■■■■■■■■■■■":GOTO100
250 PRINT:PRINT"■ENCORE 1 FOIS ■O■ OU ■N■?■";
260 GET E$: IF E$="O" THEN 18
270 IF E$<>"O" AND E$<>"N" THEN 260
```

Pour ne pas perdre la valeur précédente on fait une "permutation circulaire": ligne 210
C'est de cette manière que **tous les programmes** réalisent l'**échange.**

K ←1— J ←2— N —3→ R

```
280 END
500 REM 4 CHIFFRES AU HASARD
510 C(Y)=INT(RND(0)*10)
520 Z=81: W=1: E=1024: CO=55296: S=54272: POKE S+24,15
530 POKE E+773,Z: POKE CO+773,2
540 POKE S+1,25: POKE S,176: POKE S+5,0: POKE S+6,240
545 POKE S+4,17: FOR T=1 TO 20: NEXT: POKE S+4,16
550 FOR T=1 TO 500: NEXT: POKE E+773,32: POKE S+1,22: POKE S,227
555 POKE S+4,17: FOR T=1 TO 20: NEXT: POKE S+4,16: POKE S+1,0: POKE S,0
560 X=X+3: PRINT"■■■■■■■■■■■■■■■■■■■■■■■■"TAB(X)C(Y)"■";: POKE54273,0:Y=Y+1
570 RETURN
600 REM DANS LE VOLUME 2, CE PROGRAMME SERA ADAPTE A LA NOUVELLE
601 REM QUOTIDIENNE 4 ET LE TABLEAU DES GAINS SERA COMPLET...
610 IF C(1)=C(2) AND L(1)=L(2)ANDC(1)=L(1)ANDC(3)=L(3)THENG$="60$":GOTO650
611 IFC(2)=C(3)ANDL(2)=L(3)ANDC(2)=L(2)ANDC(1)=L(1)THENG$="60$":GOTO650
612 IF C(1)=L(2) AND C(2)=L(1) AND C(3)=L(3) THEN G$="20$": GOTO 650
613 IF C(1)=L(3) AND C(2)=L(2) AND C(3)=L(1) THEN G$="20$": GOTO650
620 IF L(2)=L(3) ANDC(2)=C(3) AND L(2)=C(2) THENG$="10$": GOTO 650
625 IF L(2)=C(2) AND L(3)=C(3) OR L(2)=C(3)ANDL(3)=C(2)THENG$="5$":GOTO 650
630 RETURN
650 PRINT"TU GAGNE "G$" TAPE ■D
655 GET D$: IF D$<>"D" THEN 655
700 REM LES GAINS
701 PRINT CHR$(14)"■■■■■■■■■■■TON CHOIX:■ ";:FOR I=0 TO 3: PRINT"■"L(I);: NEXT
702 PRINT:PRINT"■■■■■■■■■■■ORDINATEUR■ ";:FORI=0TO3:PRINT"■"C(I);:NEXT
703 PRINT:PRINT"■■■N'IMPORTE QUEL ORDRE
704 PRINT"■4 CHIFFRES DIFFERENTS
705 PRINT TAB(11)"1000$  50$": PRINT"■1 PAIRE:■   1050$ 100$"
706 PRINT"■2 PAIRES";: PRINT"■   1150$ 200$"
707 PRINT"■1 TRIPLET■  1250$ 300$": PRINT"■IDENTIQUES:■2150$
708 PRINT"■■3 DERNIERS"SPC(8)"CHIFFRES":PRINT"■DIFFERENTS:■  50$  10$
709 PRINT"   ■1 PAIRE:■  60$  20$":PRINT"■IDENTIQUES:■ 100$
710 PRINT"■■2 DERNIERS"SPC(8)"CHIFFRES":PRINT"■DIFFERENTS:■   5$   5$
715 PRINT"■IDENTIQUES:■  10$ ----":PRINT:PRINT"■■APPUIE UNE TOUCHE■";
720 GET A$: IF A$="" THEN 720
725 PRINTCHR$(142): GOTO 18
800 DATA0,0,0,0,1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4
810 DATA5,5,5,5,6,6,6,6,7,7,7,7,8,8,8,8,9,9,9,9
```

La ligne 35 **ouvre** (OPEN) un fichier d'entrer clavier (No 0) La ligne 37 **entre** (INPUT#) des données et la ligne 90 **ferme** le fichier. **Le but** de ces lignes est d'attendre une entrée des données **sans la présence du ? qui apparaît normalement après une instruction INPUT. (Volume 2)**

LE MODE TEXTE : Dans le programme ci-dessus, tu as remarqué que le tableau des gains (les lots) était écrit en lettres minuscules. Dans ce cas on dit que l'écran affiche le mode texte. Fais le chargement (LOAD) du programme: "LA QUOTIDIENNE 4" et **écris RUN 700** puis tape RETURN. L'écran affiche le tableau des gains en mode texte.

Pour obtenir le mode texte dans un programme, il suffit d'écrire: **PRINT CHR$(14)** et pour revenir au mode graphique (état normal), tu écris: **PRINT CHR$(142)**. Regarde les lignes 701 et 725 ci-dessus et tu découvriras les 2 instructions qui permettent de passer d'un mode à l'autre (codes: voir tableau page 76). **NOTE:** Tu obtiens les mêmes résultats en mode direct si tu appuies à la fois C= et SHIFT.

Programme d'un jeu de hasard: "GOBE-SOUS"

```
1 DIMC$(4,4),C1$(4): POKE 53280,5: POKE 53281,1
2 PRINT"■■■" SPC(19)" | " SPC(38)"■■■" SPC(38)"■  ■" SPC(37)"■   ■"
3 PRINT TAB(18)"■    ■" SPC(35)"■     ■" SPC(34)"■     ■" SPC(35)"■     ■"
4 PRINT TAB(15)" ▬ ■   ■ ▬" SPC(29)"■    ■ ■ ■ ■    ■"
5 PRINT TAB(13)"■      ■■■■      ■" SPC(26)"■    ▬■■■▬    ■"
6 PRINT TAB(14)"■  ■■■■■■■■  ■" SPC(29)"■■ ■▬   ▬■ ■■"
7 PRINT TAB(18)"■   ■": PRINT TAB(19)"■ ■"
8 PRINTTAB(13)"■■PROTECTION  DU"SPC(27)"CONSOMMATEUR"SPC(28)"PAR L.PICARD"
9 PRINT SPC(11)"■■APPUIE UNE TOUCHE
10 GET A$: IF A$="" THEN 10
15 FOR J= 1 TO 4
20 FOR I=1 TO 4
25 READ C$(I,J)
30 NEXT I: NEXT J
35 PRINT"■TU EMPRUNTES 5.00 $": PRINT"CHAQUE ESSAI: 25 CENTS
36 PRINT"■■TU GAGNES 100 PIECES DE 25 CENTS SI LES 4 CARTES SONT IDENTIQUES
37 PRINT"■■TU AS 1 COUP GRATUIT SI TROIS CARTES": PRINT "SONT IDENTIQUES"
38 PRINT"■■APPUIE UNE TOUCHE"
39 GETA$:IFA$=""THEN39
40 PRINT"■■":PRINTSPC(16)"0,25$"
41 PRINT"       ■r■GOBE-SOUS■h■■■"
42 PRINT"      ┌──────────■■■"
43 PRINT"      |        ■5.00■■|
44 PRINT"      | ■   ▬  ▬   ■ |
45 PRINT"      |■|| || || ||■|
46 PRINT"     ■■ ■■♦■♥■♦■♠■■■■■
47 PRINT"     ■■ ■■|| || || ||■■■■
48 PRINT"      |■ ⊔  ⊔  ⊔  ⊔■|■■
49 PRINT"      |             |■■ ■■
50 PRINT"      └─────────────┘ "
51 PRINT"        |         |
52 PRINT"      ■■            ■"
53 PRINT"■■25 CENTS POUR TOURNER LA MANIVELLE"
54 PRINT"■■APPUIE  SHIFT": VA=5: S=54272
55 B$="■■■■■■■■■■■■■■■■": CA=1: E=1024
58 WAIT 653,1: REM ATTENTE DE LA TOUCHE SHIFT
59 FOR L=1 TO 3
60 PRINTB$"■■■■■■■■■■■■■■■■"CHR$(32)CHR$(32)"■■"CHR$(32)"■■"CHR$(190);
61 POKE S+24,15
62 POKE S+5,0: POKE S+6,240: POKE S+1,38: POKE S,125: POKE S+4,33
65 FOR I=1 TO 3: PRINT"■■"CHR$(161);: NEXT I
66 PRINT"■■"CHR$(18)CHR$(160)CHR$(146)CHR$(161);
67 PRINT"■■"CHR$(32)CHR$(32)"■■"CHR$(32)"■■"CHR$(32)"■■"CHR$(144)CHR$(187);
68 PRINT"■■"CHR$(161)"■■"CHR$(161)"■■"CHR$(18)CHR$(160)CHR$(146)CHR$(161);
69 POKE S+1,19: POKE S,162: FOR T=1 TO 20: NEXT
70 PRINT"■■";: NEXT L
72 POKE S+4,32: POKE S+24,0
75 FOR K=1 TO 3
80 FOR I=1 TO 4
82 IF I=1 OR I=4 THEN Z=40: A$=B$: C=30
83 IF I= 2 THEN Z=0: A$=B$+"■": C=15
84 IF I=3 THEN Z=80: A$=B$+"■": C=15
85 ON CA GOTO 90,92,94,96
90 PRINT"■"A$C$(I,1)SPC(2)C$(I,2)SPC(2)C$(I,3)SPC(2)C$(I,4):FORT=1TOC:NEXT
91 POKE E+326+Z,32: POKE E+329+Z,32: POKE E+332+Z,32: POKE E+335+Z,32:GOTO150
92 PRINT A$C$(I,2)SPC(2)C$(I,3)SPC(2)C$(I,4): FOR T=1 TO C: NEXT
93 POKE E+329+Z,32: POKE E+332+Z,32: POKE E+335+Z,32: GOTO 150
94 PRINTA$C$(I,3)SPC(2)C$(I,4):FORT=1TOC:NEXT
```

Note: Un tableau ne doit pas être redimensionné (DIM)...autrement dit, il ne faut pas retourner au No de ligne contenant l'instruction DIM pendant l'exécution d'un programme.

```
95 POKE E+332+Z,32: POKE E+335+Z,32:GOTO150
96 PRINT A$C$(I,4): FOR T=1 TO C: NEXT
97 POKE E+335+Z,32
150 FOR T=1 TO C: NEXT
152 NEXT I: NEXT K
155 X=INT(RND(0)*4+1):C1$(CA)=C$(X,1):PRINTB$C$(X,1)
190 B$=B$+"███":CA=CA+1:IF CA=5THEN583
195 GOTO75
583 FORI=1TO4:D(I)=ASC(C1$(I)):NEXT
585 IF D(1)=D(2) AND D(1)=D(3) AND D(1)=D(4) THEN 592
586 IF D(1)=D(2) AND D(1)=D(3) THEN 610
587 IF D(1)=D(2) AND D(1)=D(4) THEN 610
588 IF D(1)=D(3) AND D(1)=D(4) THEN 610
589 IF D(2)=D(3) AND D(2)=D(4) THEN 610
590 IF G<>0 THEN 620
591 VA=VA-.25: PRINT"████████████████████"VA;: GOSUB 650: GOTO 55
592 PRINT"██TU GAGNES 100 PIECES!"SPC(6)"█25 DOLLARS":FORT=1TO500:NEXT
593 FOR T=1 TO 500: NEXT: POKE 53280,0: POKE 53281,11: FOR T=1 TO 1000:NEXT
594 PRINT"APPUIE UNE TOUCHE";
595 GET A$: IF A$="" THEN 592
597 PRINT"██CALCUL DES PROBABILITES"
598 PRINT "█1/4 * 1/4 * 1/4 * 1/4 = 1/256"
599 PRINT"██DONC 1 CHANCE SUR 256█ D'AVOIR 4 CARTES
600 PRINT"█ET 16 CHANCES SUR 256█ D'OBTENIR UN  COUP GRATUIT."
601 PRINT"██IL FAUT DEPENSER 240 (256-16) PIECES DE 25 CENTS POUR"
602 PRINT"█AVOIR UNE CHANCE DE GAGNER 100 PIECES."
603 PRINT"█CE QUI FAIT ENVIRON   40% DES PIECES
604 PRINT"RETOURNEES AUX CONSOMMATEURS █ET 60% POUR LE PATRON'!
605 PRINT"██UN CONSEIL": PRINT"ARRETE-TOI LA!":END
610 G=G+2
620 G=G-1:PRINT"█COUP(S) GRATUIT(S)"G:GOTO55
650 IF VA=.75 THEN PRINT"█ ": RETURN
655 IF VA=0 THEN PRINT"██.00█": END
660 LT=LEN(STR$(VA)): LE=LEN(STR$(INT(VA))): IF VA<1 THEN LE=1
670 IF LT-LE=2 THEN PRINT"█0"
675 IF LT-LE=0 THEN PRINT"█.00"
680 RETURN
800 DATA "♠","♥","♦","♣","♥","♦","♣","♠"
810 DATA "♦","♣","♠","♥","♣","♠","♥","♦"
```

La ligne 1 n'est pas obligatoire. Cependant, la déclaration DIM C$(4,4) fera déplacer le pointeur de "fin de tableau" **pour libérer** 288 mémoires! **D'où provient cette économie de 288 octets?**

EXPLICATIONS Les lignes 10 à 30 mettent les 16 chaînes de caractères (inscrites dans les "DATA") dans un tableau à 2 dimensions. Si tu déclares DIM C$(4,4) le C64 réservera 5 X 5 = 25 emplacements mémoires pour ton tableau (0 à 4 = **5 pour chaque dimension**). Etant donné que chaque mémoire d'un tableau de chaîne de caractères demande 3 octets, le C64 utilisera 25 X 3 = 75 octets.

Mais si tu ne mets pas l'instruction DIM, le C64 réservera automatiquement 11 (0 à 10) mémoires pour chaque dimension. Ce qui fera 11 X 11 = 121 et 121 X 3 = 363 octets. Donc en mettant DIM C$(4,4) tu économises 363 - 75 = 288 octets!

ANNEXE

Les 3 **exercices ci-dessous et l'explication (4-)** ont été oubliés pendant la transcription de ce volume 1:

EXERCICE 14-B (Voir page 93) Utilise la fonction SPC( ) pour écrire l'heure **(**heures,minutes,secondes**) au centre de l'écran.** Mets les caractères en constraste inversé.

(caractères de commande Nos 1,2 page 35)

```
10 PRINT "[][]" SPC(255),SPC(237);
20 PRINT TI$ "[][][][][][]";:GOTO 20
```

(CRSR gauche 6 fois)

EXERCICE 31-B (Voir page 100) Remplis l'écran avec 1000 petites cases bleues et introduis les caractères CRSR gauche et INST dans ton programme pour empêcher le déroulement de l'écran.

```
5 PRINT"[]";
10 FOR I=1 TO 1000
15 IF I=1000 THEN PRINTCHR$(157)CHR$(148);
20 PRINT"[]";
30 NEXT
40 GOTO 40
```

EXERCICE 32-B (Voir p.100)Utilise la fonction PEEK pour déceler la collision d'une étoile qui se déplace de haut en bas sur une ligne oblique **et** une planète qui se déplace horizontalement de gauche à droite sur une rangée choisie au hasard (3 choix).

```
10 PRINT"[]": Z=INT(RND(1)*3): E=1704+Z*40: Y=0
15 FOR X=34 TO 12 STEP -1
20 PRINT TAB(X)"[]*";: FOR T=1 TO 50:NEXT
25 IF PEEK(E+Y)=42 THEN GOSUB 100:PRINT"[]":END
30 POKE E+Y,81: FOR T=1 TO 50:NEXT
35 PRINT TAB(X)"[] ": FOR T=1 TO 25:NEXT
40 POKE E+Y,32: FOR T=1 TO 25:NEXT:Y=Y+1
50 NEXT X: GOTO 10
100 FOR I=1 TO 15
110 PRINT "[][][]";:FOR T=1 TO 20:NEXT
120 PRINT "[][][]";:FOR T=1 TO 20:NEXT
130 NEXT I: RETURN
```

4- (Voir p. 175) Comment obtenir le caractère de COMMANDE [T] (No 9 p.35):

a) Commence par **ouvrir et fermer** des guillemets.

b) Place le curseur sur le guillemet de droite et appuie **SHIFT et INST** pour éloigner les guillemets...(insertion).

c) Appuie **DEL** (sans SHIFT) et le caractère [T] apparaîtra.

MOT de la FIN: Les 180 pages réservées pour ce volume 1 sont déjà remplies! Le volume 2 te fera entrer dans le monde fascinant de la haute résolution, la programmation des caractères, les fichiers, l'écoute du monde extérieur (modem, morse), le TRI (Shell, à bulles), le langage machine!

AUREVOIR et tache de bien maîtriser ces 180 pages d'initiation!

Nous serons heureux de recevoir les commentaires, suggestions, remarques, corrections... que tu auras la gentillesse de nous adresser:
LAURIAN PICARD, COLLEGE D'ARTHABASKA, 905 BOIS-FRANCS SUD, ARTHABASKA, G6P 5W1 ou 2648 MOREAU, MONTREAL, H1W 2M8